Framework for Securing and Querying Healthcare Data using Blockchain

A Thesis

submitted in fulfilment of the requirements for the award of degree of

Doctor of Philosophy

Submitted By

Jasleen Kaur

(Roll No.: 901803009)

Under the guidance of

Dr. Rinkle Rani

(Professor, DCSE)

Dr. Nidhi Kalra

(Assistant Professor, DCSE)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY

PATIALA - 147004, Punjab

October 2024

DEDICATED WITH EXTREME AFFECTION AND GRATITUDE TO

*My **parents** Mr. RavinderPal Singh Bedi and Mrs. Gursharanjit Kaur Bedi,*

*My **sister** Ishpreet Kaur, and*

*My **supervisor(s)** Dr. Rinkle Rani and Dr. Nidhi Kalra*

Certificate

I hereby certify that the work which is presented in this thesis entitled "**Framework for Securing and Querying Healthcare Data using Blockchain**", in fulfilment of the requirement for the award of degree of "**Doctor of Philosophy**" submitted in Department of Computer Science and Engineering of Thapar Institute of Engineering & Technology, Patiala, is an authentic record of my own work carried out under the supervision of **Dr. Rinkle Rani** and **Dr. Nidhi Kalra** refers other research works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

Jasleen Kaur
(Jasleen Kaur)
Regn. No. 901803009

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Rinkle 22/10/24
(Dr. Rinkle Rani)
Professor

Nkalra 22/10/24
(Dr. Nidhi Kalra)
Assistant Professor

Department of Computer Science & Engineering
Thapar Institute of Engineering and Technology, Patiala

Acknowledgement

First, I would like to thank the Almighty God, who gave me the strength and courage to overcome all the obstacles and complete my PhD journey.

Further, I would like to express my warmest gratitude to my supervisors, **Dr. Rinkle Rani** and **Dr. Nidhi Kalra**, for their excellent guidance, immense support and motivation throughout my research work. Their positive mindset, strength, and words of encouragement have always pushed me to work harder for the entire duration of the research work. The optimistic lessons that I have learned from them will stay with me forever for good.

My sincere thanks to **Dr. Shalini Batra**, Professor and Head, DCSE, Thapar Institute of Engineering and Technology, Patiala, for providing me the necessary administrative assistance and infrastructure that helped me in the completion of my research work. I am thankful to my doctoral committee members: **Dr. Rajkumar Tekchandani**, **Dr. Sanmeet Kaur**, and **Dr. Neeru Jindal** for their insightful suggestions and regular monitoring of the research work. I am thankful to all the faculty, staff members and research scholars of DCSE for their support.

I would like to offer my heartfelt thanks to my father, **Mr. RavinderPal Singh Bedi**, for his life lessons and kind words that gave me the sense of encouragement and confidence throughout this journey. I am thankful to my mother **Mrs. Gursharanjit Kaur Bedi**, my confidante, whose love and care have kept me going. Also, I express my thanks to my sister, **Ishpreet Kaur**, for her constant support, suggestions and confidence in me. I also acknowledge the cooperation and encouragement by my friends all this while.

Jasleen Kaur
(Jasleen Kaur)

Patiala

October, 2024

Abstract

Technological advancements are significantly transforming operations within organizations. When it comes to the healthcare sector, the development rate is touching new heights and being supported by cutting-edge technology as the whole system moves towards a patient-centred approach. Healthcare service providers have begun to use Electronic Health Record (EHR) management systems to overcome issues associated with traditional, manual, paper-based, outdated systems that can make the diagnosis time-consuming and complicated for the doctor. EHRs, primarily used to improve health-related information management, are frequently approached for a secondary purpose by other medical/non-medical institutions or Third-Party Administrators/Alliance (TPAs) such as insurance companies and data analysts. However, the existing EHR systems are centralized and always at risk of a single point of failure. With exponential growth in size of EHRs, scalability issues arise, making the systems more susceptible to cyber-attacks that risk patient privacy and data integrity.

To address these challenges, decentralized solutions are required. Blockchain technology, with its decentralized and cryptographic behaviour, provides a powerful solution to these issues. It ensures data security, integrity, confidentiality, and non-repudiation, making it an ideal candidate for healthcare applications.

This research proposes a decentralized permission-based blockchain framework for the secure sharing and querying of healthcare data. The framework is divided into three stages: During first stage, a permissioned blockchain architecture involving multiple healthcare entities for transparent, trusted, and immutable storage, sharing, and querying of EHR data is proposed and implemented. Further, for better data accessibility and security, the algorithms in terms of smart contracts are defined. In the existing approaches, LevelDB (key store database) is used as a state database that is insufficient for handling complex queries. To overcome

this issue, this research utilized CouchDB (document store database) as a state database, enabling searches to be conducted using both keys and data values rather than just keys alone. Further, "Hyperledger Caliper", a benchmarking tool, is utilized to evaluate the various performance indicators for this framework, such as transaction throughput and latency. The system is also evaluated for handling complex queries with examples.

In the second stage, the Identity-Based Proxy Re-Encryption (IB-PRE) algorithm is utilized to improve the privacy and security of the EHR data. In addition to this, to overcome scalability issues and achieve data integrity in managing EHRs, an Interplanetary Distributed File System (IPFS), a distributed off-chain storage, is employed. It is a content-addressable storage that ensures the integrity of the content such that a slight modification in the stored EHR records results in a change in the obtained hash value.

In the third stage, the framework implements a dual-channel blockchain architecture combined with two cryptographic algorithms, i.e. Rivest-Shamir-Adleman (RSA) and Advanced Encryption Standard (AES), to provide the security and rapid retrieval of healthcare information. Furthermore, the concept of private data collection is incorporated to securely store confidential patient information, guaranteeing privacy, security and limited access. Also, an Access Control List (ACL) is defined for different users to implement access permissions, i.e., grant and revoke access to viewers while sharing information. The proposed framework is tested and validated by conducting a qualitative assessment of the designed contracts to ensure proper functionality and interaction among various components. The system performance is evaluated by considering different scenarios, including varied transaction loads, transaction types, number of channels, etc. Experimental results demonstrate that throughput increases significantly at a specific transaction rate when using dual channel, making the system more efficient compared to a single channel.

Additionally, latency drops considerably, leading to a notable improvement in system performance. The proposed framework's security analysis is also conducted through two vulnerability assessment tools. This work also compares the proposed framework's effectiveness and efficiency with other related works. The evaluation metrics include security attack resistance, features offered, scalability, transaction throughput, and latency. The proposed framework is intended to strictly enforce security and privacy standards while also improving data management and accessibility in healthcare applications.

Keywords: Blockchain, Dual-Channel, EHR, Smart Contracts, Hyperledger Fabric, Private Data Collection, RSA, and AES

Contents

List of Abbreviations

Abbreviation	Description
ACL:	Access Control List
AES:	Advanced Encryption Standard
BFT:	Byzantine Fault Tolerance
CA:	Central Authority
EHR:	Electronic Health Records
HIPAA:	Health Insurance Portability and Accountability Act
HLF:	Hyperledger Fabric
IB-PRE:	Identity-Based Proxy Re-Encryption
IPFS:	InterPlatenary File System
MSP:	Membership Service Provider
P2P:	Peer-to-Peer
PBFT:	Practical Byzantine Fault Tolerance
PoS:	Proof of Stake
PoW:	Proof-of-Work
RSA:	Rivest-Shamir-Adleman
SC:	Smart Contracts
TPA:	Third Party Administrators/Alliance

List of Figures

List of Tables

List of Algorithms

Chapter 1

INTRODUCTION

This chapter begins by outlining the motivation behind the research work. It then introduces the fundamental concepts of blockchain technology, including its various types, platforms, and tools. Finally, the thesis organization is presented in the last section of this chapter.

The impact of Information Technology (IT) on the healthcare sector has been transformative, driving significant improvements in the quality, efficiency, and availability of healthcare services. It has significantly improved various services and aspects of the healthcare sector, such as appointment scheduling, remote consultation and monitoring, secure storage and retrieval of patient data, lab and medical records tracking, emergency care, etc. Despite technological advancements in the healthcare sector, many hospitals and medical institutes still use paper-based health medical records that are easy to lose or misplace. Health Records are valuable assets for every patient. A health record consists of the medical history of a patient, such as disease information, prescriptions, lab test reports, vitals (body temperature, pulse rate, blood pressure), financial information (medical bills, debit/credit card, and bank account details) along with personal information such as name, age, and physical signs, including height and weight. A long history of records has to be maintained by a person suffering from a critical disease that is essential for his treatment. Maintaining a long history of paper-based health records is a cumbrous process. These systems suffer from numerous drawbacks, including excessive physical storage requirements, limited security measures, time-consuming and error-prone processes. These

systems exhibit inefficiency, lack organization, display inconsistency, and are not tamper-resistant. Complete and accurate medical data is a valuable asset that must be securely preserved and shared. Though it should be kept personal, sharing becomes inevitable so that appropriate treatment can be given. Physical health records are difficult to coordinate among various stakeholders and prone to loss or harm.

To address the aforementioned concerns, the evolution of the Electronic Health Record (EHR) system is a critical advancement in the field of modern healthcare. EHRs are much more dynamic and easier to manage, facilitate, and share among stakeholders such as doctors, patients, lab technicians, etc. These systems have been utilized by several hospitals, medical institutes, and dispensaries to store and manage patients' medical data. The report shows a rise in accepting EHR systems from 9.4% to 83.8% across non-federal acute care hospitals from 2008 to 2015 [1]. The American Hospital Association (AHA) found that approx. 96% of non-federal acute care hospitals and nearly four out of five (approx. 78%) office-based physicians have adopted certified EHR systems in a subsequent survey done in 2021. This is a significant improvement over 2011, when only 28% of hospitals and 34% of physicians used EHR systems [2]. Compared to traditional, manual, paper-based medical records, EHR systems are more efficient, organized, and less error-prone.

With the exponential growth in the size of EHRs, scalability issues arise, making the systems more susceptible to cyber-attacks that risk patient privacy and data integrity. Therefore, these challenges necessitate the development of decentralized solutions integrating the latest technologies to provide data immutability and safeguards against unauthorized access, facilitating efficient and secure data sharing among healthcare providers. A detailed description of the various issues that motivate this research is discussed below in the motivation section.

1.1 Motivation

Hospitals are the primary administrators and controllers of EHR systems, which are mostly centralized. There are strict regulations and procedures for transferring sensitive data, such as medical records, outside the institute[3]. For proper diagnosis and timely medication, the doctor requires the complete history of the patient. It is not convenient to access fragmented records in case of emergency. Moreover, a person has to undergo the same medical test that was previously performed at another hospital if medical history is not maintained.

Interoperability issues arise because the medical data is dispersed across multiple hospitals rather than being centralized, and a variety of data standards are used by different hospitals. This hinders the transfer of medical data across various hospitals, adversely affecting the patient's care and making the system inefficient. Individuals use health insurance policies to safeguard against future health uncertainties. However, when they request to process an eligible claim, processing becomes time-consuming and cumbersome due to various medical records-related issues. Even for the insurance providers, this increases processing costs due to incomplete medical records. Sometimes, while taking health insurance policies, many people suppress or don't disclose vital information related to their health history. Also, the middleman, agent, or person filling the forms falsifies the records knowingly or unknowingly, which results in the rejection of claims and legal disputes. For health insurance policies, Health Insurance providers either tie up with an external Third Party Administrator/Alliance (TPA) or opt for in-house TPA. This is because the whole process requires complete & accurate medical records that involve different stakeholders such as patients, doctors, and hospitals, which consume a lot of administrative cost and time, resulting in patient dissatisfaction[4]. Therefore, solutions for sharing accurate records with TPA or other service providers are required to speed up the claiming process, enhancing the overall system's effectiveness and efficiency.

Medical information is always susceptible to various security threats, including data loss, leaks, malicious alterations, and other concerns[5]. According to statistics released by HIPAA, 37.47% more medical records were compromised in 2019 compared to 2018[6]. According to statistics for 2024, the US healthcare business experienced an annual loss of around $7 billion due to stolen Protected Health Information (PHI). Healthcare data breaches incur the most significant expenses across all industries, amounting to $408 per record[7]. To tackle the above issues, researchers have proposed various schemes. The authors[8] discussed the adoption of the next-generation techniques in healthcare 4.0 systems. In[9], the authors proposed a personal health record-sharing model using an attribute-based privacy-aware in a cloud-based environment. Similarly, the authors proposed a privacy-preserving authentication scheme to monitor real-time medical systems[10]. However, preserving confidential medical data at third-party premises is always at risk. Suggestions related to the EHRs privacy and security are suggested to both healthcare and cloud service providers[11]. Similarly, the authors[12] combine blockchain with AI for scheduling tasks while enduring security and user privacy and optimising edge computing in IoT-based environments. Also, AI models such as transformers revolutionize the execution of automated tasks. However, these are prone to many biases that lead to privacy-related issues. Thus, the authors proposed the T5 approach (Text-to-Text-Transfer-Transformer model) to estimate and eliminate gender bias in contextualized embeddings[13].

Additionally, EHR primarily comprises unstructured data, including patient clinical notes, X-ray images, and lab test reports, which are available in many formats. According to a report[14], 80% of the medical data is unstructured. The utilization of blockchain technology for secure storage, convenient access, and efficient query processing to enable quick retrieval of data presents a significant challenge. Different solutions are provided by various authors[11,15,16] using off-chain storage, cloud-based storage with query support in different blockchain platforms.

Also, user preferences in information retrieval play a vital role in healthcare[17]. Still, there is a need for a NoSQL database to manage unstructured healthcare data and ensure secure storage efficiently. Moreover, various healthcare organizations traditionally obtained health information by exchanging and transmitting medical data. Examples include the Blue Button Connector, a project lead by the U.S. government, Apple's mobile healthcare application, and Google Health (https://health.google.com/health/). Nevertheless, these solutions have not yet met the criteria for an optimal healthcare information system, including aspects such as security, reliability, and transparency. Therefore, a management system characterized by decentralization, verifiability, and immutability is essential for handling medical data. Thus, the trend shifts to decentralized technologies such as blockchain, which is gaining popularity nowadays and widely adopted in different areas[18]. Blockchain is a distributed ledger with the attractive properties of immutability, tamper-resistance, and traceability of the log, which records all CRUD operations performed on the data as transactions. Cryptographically linked blocks store transactions to create a blockchain. Furthermore, the advent of diverse blockchain platforms such as Ethereum, Hyperledger Fabric, Sawtooth, and smart contracts[19] has enhanced the functionality of blockchain, leading to its widespread application in numerous industries, particularly in healthcare[20].

Most existing studies about the blockchain-based healthcare system focus on decentralized apps (DApp), smart contracts, and permission-less networks. They use a public blockchain-based platform, i.e., Ethereum, to access and share EHRs securely. However, EHRs contain the patient's confidential data that needs to be frequently stored, queried, and securely shared with other healthcare entities. However, in a permission-less network, the entities are anonymous, which makes the system untrusted. Furthermore, medical data exhibits heterogeneity and fragmentation across various healthcare institutions and physicians. Obtaining comprehensive medical records and effectively addressing queries is crucial to preventing treatment delays and enhancing the quality of healthcare services. Lack

of consideration of several performance measures, such as scalability, throughput, latency, CPU, and memory usage, while designing a blockchain-based EHR system. Various researchers have provided many solutions to the above issues, yet the requirements are not satisfied. Hence, a permissions-based framework with a decentralized, tamper-proof, and immutable healthcare network is required to address the aforementioned challenges.

Therefore, motivated by the above-discussed issues, this work aims to develop a permissioned blockchain-based framework for the secure storage, retrieval, and distribution of healthcare data among authorized entities. Additionally, it enables patients to have full ownership of their assets so that they have complete control over whom, when, and what they share utilizing blockchain technology. They can observe all the activities anywhere, anytime, instead of relying on any centralized infrastructure.

1.2 Fundamentals of Blockchain Technology

In 1991, Stuart Haber and W. Scott Stornetta envisioned the concept that has been widely recognized as a Blockchain [21]. Their initial project was developing a cryptographically secured chain of blocks that prevented anyone from tampering with document timestamps. In 1992, they expanded their system to include Merkle trees which improved efficiency and permitted the accumulation of additional documents on a single block. In 2008, Blockchain history started to gain usefulness when Satoshi Nakamoto conceptualized the first Blockchain by releasing the first whitepaper [22] "Bitcoin: A peer-to-peer electronic cash system" as the underlying technology of Bitcoin in 2009. The authors of this research elaborated on how the technology is effectively designed to strengthen digital trust due to its decentralized nature, ensuring that no single entity would hold control over any aspect of it. Over time, this digital ledger technology has developed, leading to the emergence of new applications that constitute the

historical narrative of blockchain illustrated in Figure 1.1

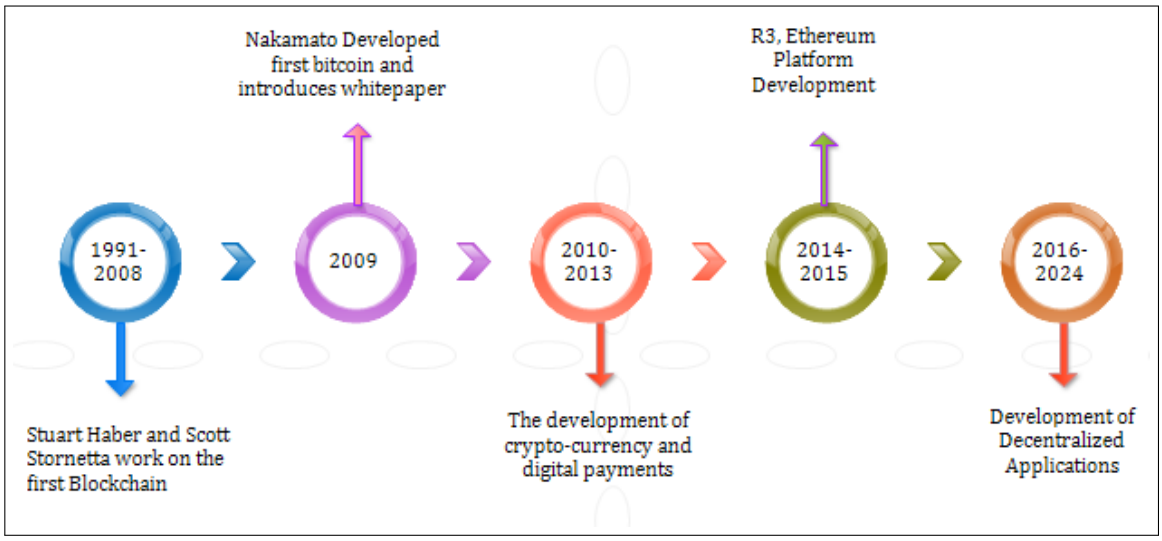


Figure 1.1: History of Blockchain

Blockchain is *"A decentralized computational and information sharing platform that enables multiple authoritative domains which do not trust each other, to cooperate, coordinate and collaborate in a rational decision-making process"*[23]. Another formal definition of Blockchain is defined by Macro et al. as "An open distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way"[24].

1.2.1 Blockchain

Blockchain is a secure and decentralized system of keeping track of data, where records cannot be altered and are managed by a network of computers rather than a central authority. The origins of this can be traced back to the research efforts of Haber and Stornetta[25]. Blockchain technology incorporates cryptographic hash algorithms such as MD5, SHA256, and SHA512, as well as Merkle tree[26]. The block structure and cryptographically linked blocks that make the chain tamper-proof are illustrated in Figure 1.2.

Block header and additional metadata, including the block size, transaction counter, and transaction information, create the block. The block header contains

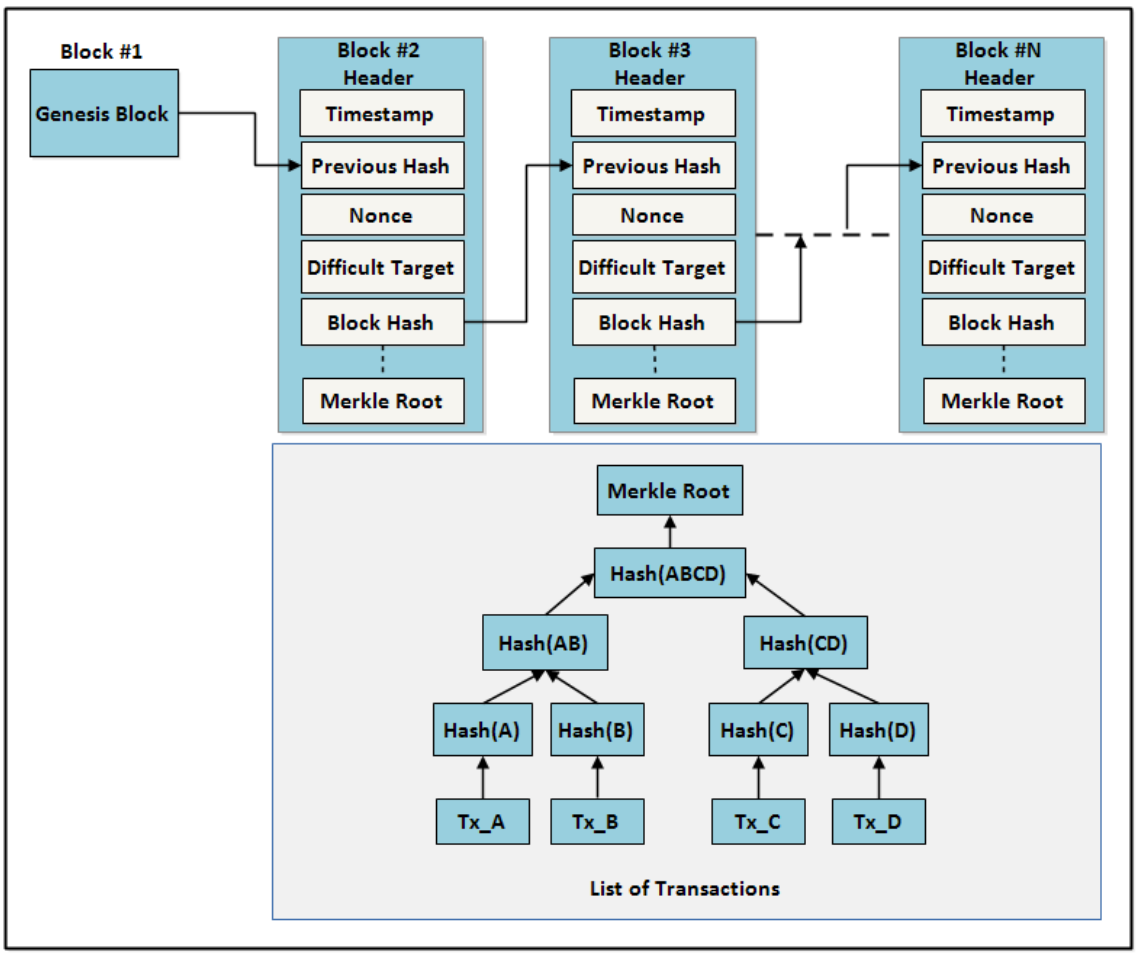


Figure 1.2: Blockchain Structure and Cryptographically Linked Chain of Blocks

the following fields that act as messages or data strings and are used for calculating the hash of the current block. Genesis Block in the chain is the first block.

- Version: It is a 4 byte field used to track policies and software updates in a block.
- TimeStamp: It denotes the creation time of the block, which is of 4 bytes.
- Nonce: It stands for Number Used Only Once. It is a 4-byte field added to a hashed or encrypted block in a blockchain after the mining process meets the difficulty level of restriction.
- Previous Hash: It is a 4-byte field that holds the hash of the previous block.
- Difficult Target: It is also a 4-byte field used to set difficulty levels during

mining.

- Merkle Root: This is a 32-byte field that denotes the root hash value of the Merkle tree containing the transactions of the current block.

The encrypted value of block header BH is known as Hash of the current block B_h shown in Equation 1.1.

$$B_h = SHA_{256} \tag{1.1}$$

1.2.2 Smart Contracts

Smart contracts (SCs) are computer programs that are executed automatically on the blockchain. In these programs, business logic and access control policies are defined, and their accurate execution is enforced by consensus protocol. As SCs are embedded in the blockchain, they offer both immutability and transparency in blockchain networks. The Ethereum platform makes use of Solidity programming language to write contracts. In Hyperledger Fabric, the terms chain codes and SCs are used interchangeably. Coding languages such as GO, Java etc., can be used to write chain code in the Hyperledger Fabric platform.

1.2.3 Consensus

The consensus is a method of reaching to a mutual agreement regarding the present status of the distributed ledger through all nodes in the Blockchain Network (BN). Thus, it achieves trust among unknown peers and reliability in the blockchain. Table 1.1 presents the comparison among various consensus algorithms. There are few algorithms which are energy efficient, whereas others consume more power and resources.

The Proof of Work (PoW) algorithm is an energy-intensive algorithm used by the Bitcoin and Ethereum platforms. It requires high computational resources, resulting in high energy consumption and slow transaction processing. On the

other hand, Proof of Stake (PoS) is employed in Ethereum 2.0 and Cardano as an energy-efficient consensus mechanism. It relies on validators' stake, reducing resource consumption but potentially leading to centralization. Subsequently, an alternative version of PoS, known as Delegated PoS, is introduced, wherein chosen delegates are employed to achieve consensus. Hyperledger Fabric is utilized for permissioned networks, where the Practical Byzantine Fault Tolerance (PBFT) algorithm is used. However, PBFT faces challenges in terms of scalability despite its reasonable efficiency. Furthermore, Raft is an alternative consensus technique that can be employed in Hyperledger Fabric. Tendermint is another protocol that integrates both PoS and BFT. It is known for its efficiency. Similarly, the consensus protocols Hashgraphs by Hedera and Tendermint by Cosmos platform are highly efficient. However, they suffer from a complex configuration process and licensing constraints.

Table 1.1: Comparison of Various Consensus Algorithms

Consensus Algorithm	**Energy Efficient**	**Resource Consumption**	**Limitation(s)**	**Example(s)**
PoW [27]	Less	High resource consumption	Consumes more energy, less security and low throughput.	Bitcoin, Litecoin, Ethereum (pre 2.0)
BFT [27]	Medium	High CPU Consumption	Semi-trusted, less Scalable	Hyperledger Fabric
PBFT [28]	Medium	High bandwidth consumption	More nodes leads to communication overhead.	Hyperledger Fabric
PoS [27]	High	Low	Consensus paid to highest paid stakeholders,less security	Ethereum, Peercoin
PoET [28]	High	Medium	Complex Setup	HyperLedger Sawtooth
Tendermint [29]	High	Low	The cost of mining is high	Tendermint, Cosmos
PoA	High	Low(trusted validators)	Centralization, trust issues	Vechain
Raft	High	Low(leader-Based)	Scalability issues	Hyperledger Fabric
Hashgraph [30]	High	Low	Licensing constraints	Hedera

1.3 Types of Blockchain

Blockchain has evolved as a versatile tool that is utilized as a solution in multiple applications across various sectors. Various types of blockchains are employed based on different use cases and their specific challenges, as depicted Figure 1.3. Blockchain is basically categorized into four categories: public/permissionless, private/permissioned, consortium, and hybrid type. In a public blockchain, anyone can join the network, send, access, receive, and verify transactions in the network. The single authority does not own the network, thus making it decentralized. *Ethereum* is a popular example of a public blockchain. The private blockchain is alternatively stated as the permissioned blockchain. In this particular blockchain type, the network is owned by one or more entities providing enhanced privacy and restricted access, rendering them optimal for internal enterprise use. The proposed framework utilizes *Hyperledger Fabric*, a renowned private blockchain developed by the Linux Foundation.

The third type is a consortium blockchain governed by a group of organizations. Designated nodes managed the consensus process in this system. Further, the transactions are initiated, communicated and validated by the validator node, whereas other nodes accept or process them. R3 and Quorum are instances of consortium-type blockchains.

The last one is a hybrid blockchain that combines the features of both public and private blockchain. It enables organizations to establish a private, permission-based model with a public, permissionless system, granting them control over access to specific data stored on the blockchain. Ripple and Komodo are examples of hybrid blockchains.

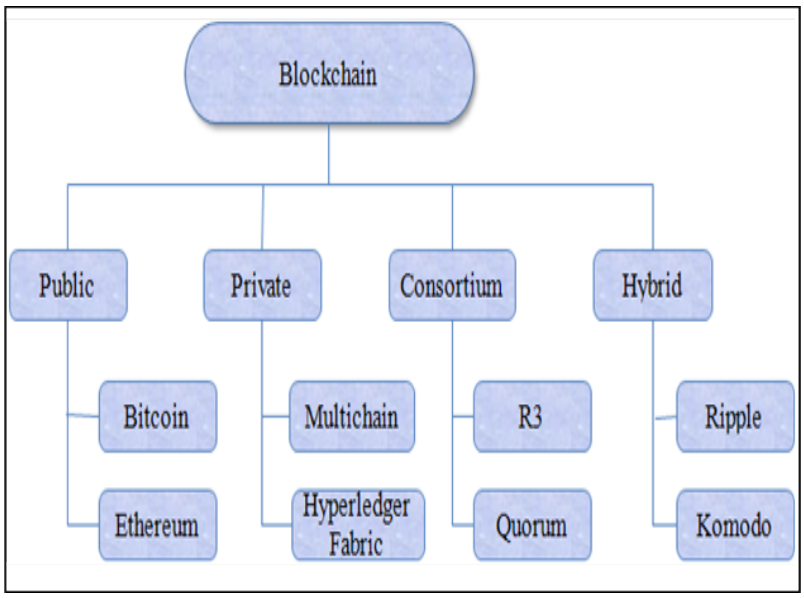


Figure 1.3: Blockchain Categories

1.4 Blockchain Platforms and Tools

There are numerous platforms and tools that can be utilized to construct blockchain-based applications. Several frequently used platforms and tools are discussed in this section and displayed in Figure 1.4.

1.4.1 Blockchain Platforms

There are various Blockchain platforms available that many enterprises use for building blockchain-based applications other than Bitcoin. Following are some of the popular Blockchain platforms:

1. Ethereum: It is a decentralized blockchain platform that enables the creation and operation of SCs and decentralized applications (dapps). It is a distributed blockchain network that was officially presented in 2015 by Vitalik Buterin[31]. The authors employed the blockchain concept, which was recently used in the well-known digital currency Bitcoin[32] by Santoshi Nakamoto. The thought behind Ethereum was to design a trustless system using smart contracts that would be open source and have

a feature for programmable blockchain. Solidity language is utilized to customize blockchain and implement SCs. These contracts are autonomous and immutable programs for verifying and executing an agreement's terms or set of rules. They are autonomous, thus decreasing the cost of creating and maintaining a centralized database.

2. Hyperledger Fabric: It is a permissioned, pluggable, open-source blockchain technology founded in 2015 by the Linux Foundation. The platform has a modular architecture, is simple to use, and supports various peers, SCs, adjustable membership, and consensus services. In HLF, peer nodes execute chain code functions, access ledger data, and approve transactions. Orderer nodes send endorsed transactions to the rest of the network's peers, ensuring blockchain network integrity.

3. BigchainDB: An open-source platform that combines distributed database features and blockchain. It supports both public and private networks. In this platform, the federation of nodes records each data without requiring a Merkle tree or sidechains.

4. Stellar: It is a public blockchain platform that offers various tools for developers, enabling them to create portable and mobile wallets that feel like real cash rather than cryptocurrency. The network built using Stellar outperforms most blockchain-based systems in terms of speed, cost, and energy efficiency. To increase the process for cross-border funds transactions, ICICI Bank and RippleFox employ the Ripple blockchain network.

5. R3 Corda: An open-source distributed ledger launched in 2015, which empowers organizations to execute legitimately with SCs. R3 Corda operates on an authorization framework that permits only authorized parties to access specific data rather than the entire network. It boosts privacy levels by implementing access control for digital records. Financial institutions such

as HSBC and Bank of America are heavily investing in R3 Corda due to its detailed control over digital records.

Figure 1.4: Blockchain Platforms and Tools

1.4.2 Blockchain Tools

The advancement of blockchain technology has resulted in the invention of numerous blockchain tools that attempt to facilitate the development of apps based on blockchain in a simplified manner. Some of the widely used tools are discussed below:

1. Remix IDE: It is a user-friendly web-based tool designed to create and deploy smart contracts. It also provides in-built support to debug, test, and deploy the smart contracts created using the Solidity programming language [33].

2. Truffle Framework: Truffle Framework is a comprehensive toolset that provides an environment for both developing and testing specifically tailored for creating applications based on the Ethereum blockchain. It also offers automation testing using Mocha and Chai tools [34].

3. Solium: Security plays a crucial role while designing blockchain apps. It is essential to ensure that the Solidity code is devoid of any security

vulnerabilities. Solium tool is specially developed to improve Solidity code and address vulnerabilities wiin the code. To use Solium, install it by using the *npm* command [35].

4. Geth: It is an Ethereum client developed in the Go programming language, utilized for operating Ethereum nodes. It provides the ability to mine ether tokens, develop SCs, transfer tokens, and investigate the block history.

5. Ganache: It is a private Ethereum blockchain utilized for conducting tests, executing commands, and inspecting the state of the blockchain while managing its operational parameters. It also offers a graphical interface for managing blockchain.

6. MetaMask: It is a browser extension that acts as a gateway to access blockchain apps and provides a cryptocurrency wallet. It enables the execution of Ethereum decentralized applications (dApps) directly in the web browser without the need for a complete Ethereum node. MetaMask contains a highly secure identity vault that offers a user-friendly interface for managing identities across many websites and digitally signing transactions.

1.5 Thesis Organization

The thesis is organized as per the following chapters:

Chapter 1: Introduction

This chapter begins by outlining the motivation behind the research work. It then introduces the fundamental concepts of blockchain technology, including its various types, platforms, and tools. At last, the chapter concludes with a summary and overview of the thesis organization. This chapter partially addresses Research Objective 1.

Chapter 2: Literature Review

This chapter provides an in-depth review of the literature on various application areas utilizing blockchain technology. It also explores different querying techniques to handle vast amounts of healthcare data, as well as blockchain-based EHR/EMR systems, to tackle the significant challenges in data security, privacy, and interoperability, which are prevalent in current healthcare data management practices. Research Objective 1 is accomplished in this chapter.

Chapter 3: Problem Formulation

This chapter presents the research gaps identified through an extensive literature review. On the basis of these research gaps, three research objectives have been identified focusing on developing a blockchain-based framework to manage healthcare data. These research gaps are systematically addressed in the upcoming chapters of the thesis.

Chapter 4: Blockchain based Framework for Secured Storage, Sharing and Querying of Healthcare Records

In this chapter, a permissioned blockchain-based framework involving multiple healthcare entities to store, share, and query EHR data is designed and implemented. The algorithms in terms of smart contracts are designed to define the functionality of patients, doctors, and Third-Party Administrators (TPA). Further, the experimental analysis of the proposed framework is conducted, and the performance of the proposed approach is evaluated using standard performance parameters that demonstrate its effectiveness and efficiency. This chapter partially accomplishes the research objectives 2 and 3.

Chapter 5: An Approach for Enhancing Security and Privacy of Healthcare Data

This chapter describes the Identity-Based Proxy Re-Encryption (IB-PRE) algorithm, which is utilized to enhance the privacy and security of healthcare

data. The integration of the IB-PRE algorithm into a permissioned blockchain-based framework for healthcare data ensures the secure sharing of records while preserving privacy by avoiding the exposure of private keys. Several experiments on real-time medical datasets to validate and assess the performance of the proposed framework using distributed storage are conducted. The research objectives 2 and 3 are partially achieved in this chapter.

Chapter 6: Dual Channel Blockchain based Framework for Healthcare Data

This chapter upgrades the framework by developing dual-channel blockchain architecture combined with two cryptographic algorithms, i.e. Rivest-Shamir-Adleman (RSA) and Advanced Encryption Standard (AES), to provide enhanced security and rapid retrieval of healthcare information. The proposed approach includes the features of Private Data Collection and Access Control Lists (ACL) to communicate privately and securely. This chapter provides a detailed experimental evaluation, including a functionality test, security analysis, performance measurement of the proposed framework, and a comparison with other existing works. Lastly, it concludes that the proposed strategy is highly capable of maintaining security and privacy standards while ensuring efficient management and accessibility of data in healthcare applications. This chapter succeeds in achieving objectives 2 and 3.

Chapter 7: Conclusion and Future Directions

This chapter concludes the thesis by presenting a concise overview of the proposed blockchain-based framework designed to secure and facilitate the querying of healthcare data. Additionally, it discusses some open issues that can be explored in future research.

Chapter 2

LITERATURE REVIEW

This chapter presents a comprehensive overview of numerous application domains utilizing blockchain technology, various querying techniques for blockchain systems, and blockchain-based EHR systems. Section 2.1 contains a summary of research related to the usage of blockchain in various application areas. Section 2.2 compares the existing storage and querying techniques for blockchain systems. Section 2.3 comprises the summary of research work on blockchain-based EHR systems.

2.1 Usage of Blockchain in Various Application Areas

This section aims to discuss some potential Blockchain enabled applications that offer advantages over traditional systems through a comprehensive literature analysis. Initially popularized by the global bitcoin industry, blockchain technology has also been actively used in other sectors. The blockchain system has made it easier to design simplified methods for identification and authentication. These methods are safe and secure as they lie outside the control of any government or non-governmental organization and cannot be tampered with. The summary of various application domains that utilize blockchain technology is described in Table 2.1

Cryptocurrency

Blockchain technology was initially developed for digital cryptocurrencies, i.e., Bitcoin[32]. Owing to its transparent, immutable, decentralized, and secure nature, it has become a revolutionary technology that is utilized to develop solutions for other industries. In digital currency and other financial applications, this technology provides safe and transparent transactions[36] but suffers from high energy consumption. Zhou et al.[37] discuss the benefits and risks associated with decentralized finance. Decentralized Finance (DeFi) offers direct user-to-user services, including lending, borrowing, and trading activities. The middleman in these activities are no longer required, reducing the overall cost and speeding up the process. Catalini and Gans[38] explore the economic implications of the blockchain, highlighting the areas of research that need further investigation, including the ability to scale, regulatory considerations, and energy usage.

Supply Chain Management

Supply Chain Management involves real-time tracking of goods' origin, status, and movement. These chains are vulnerable to adversarial attacks. The research investigates supply chain attacks, focusing on successful examples and examining the possibilities for EU and national laws to prohibit or punish companies that fail to mitigate them[39]. The authors of[40,41] discuss blockchain integration with the supply chain, provide theoretical research on the importance of integrating both fields. This integration enhances transparency & traceability and reduces fraudulent activities. Schoner[42] and Feng[43] utilize this combination in the pharmaceuticals and agri-food sectors. In their work, the authors[44] design a system where symmetric and asymmetric encryption is used, such that the product owner can trace their product and transfer ownership quickly with improved traceability and privacy of the system. Min[45] also discusses how blockchain innovation improves supply chain resilience during increased risks and uncertainty. Various researchers critically analyze the combination of blockchain

and AI for the supply chain[46].

Healthcare

As the technology improves and blockchain becomes more prevalent, it is widely used for various healthcare applications, including managing EHR/PHR data, keeping genomics data safe, preventing healthcare fraudulent activities, performing clinical trials, and telemedicine[47,48,49,50] In 2016, Azaria et al.[51] introduced a blockchain-driven approach called MedRec for healthcare records management. Further, the studies conducted by Gordon and Catalini[52] and Mettler[53] highlight how this technology might improve data security and safeguard patient privacy. Omar et al. in 2019 suggested a platform to provide patients with complete control of data placed in the cloud and to attain privacy through a permissioned blockchain built on Ethereum[54]. However, some issues have been pointed out in research that needs to be focused such as system scalability, data standardization, and user privacy.

Voting System

Sierra Leone, one of the countries in West Africa, held its first government elections using an E-voting system in 2018 based on blockchain technology. The main reason for using this system is to decrease voting costs by eliminating paper polls and to prevent corruption in the voting process. The process is such that the voters anonymously cast votes in an immutable blockchain ledger and provide instant results after the election[55]. Similarly, the authors[36] in their work, review the requirements of the current voting system and then propose a model named Votereum. This system based on blockchain utilizes one server to oversee the whole system, while another server is dedicated to managing blockchain-related queries. In another work, a model is introduced by the authors based on an Android application in which security is enhanced by embedding proper authentication and authorization of voters. A unique ID is used for

authentication, a biometric for authorization, and a one-time password for voter verification. The cast votes are stored as transactions in the blockchain to keep track of polls, and for security, a 128-bit AES encryption algorithm is used [56].

Education

Academic certificates are valuable assets as they reflect the personal qualifications of their holders. Education credentials are used as safe passage for immigration, Visa, Employment and Promotions. Blockchain technology has immense ability to transform the education system because of its unique features, such as data privacy, decentralization of the network, and immutable and transparent logs. In 2015, the authors developed Blockcert at MIT Media Lab allowing users to create, issue, examine, and validate credentials via the Bitcoin network. On this platform, issuing a certificate is fairly simple and only requires the recipient's blockchain address, the issuer's public key, and certificate issue date [57]. After the success of Blockcert, the University of Nicosia (UNIC) became the first higher education system to adopt Blockchain technology for the distribution of academic certificates, and Malta became the first European Country to follow the lead. Its usage will provide data privacy, real-time, automated verification from anywhere in the country, a tamper-proof and fraud-resistant system, and eliminate dependence on the issuing authority for future verifications, [58].

Real Estate

Integrating blockchain technology in real estate is an open-source, public network with no central authority. It functions as a distributed database or a public registry storing assets and transaction information over P2P networks. The data is placed in all of the network's participant nodes. The decentralized standard system for real estate records would eliminate frauds relating to property documents and the requirement of a middleman or agent, shortening the process, duration, and cost involved. It would also minimize the need for a physical visit to the property site

for verification while improving the process efficiency and trustworthiness among transacting parties. The authors of[59,60] emphasize the potential advantages and difficulties. It mitigates fraudulent activities and enhances the efficiency of transactions. However, this domain faces challenges regarding legal recognition and system integration.

Energy

Blockchain facilitates P2P energy trading and improves grid resilience by providing a transparent and secure platform for energy transactions. Andoni et al.[61] and Mengelkamp et al.[62] explore applications in energy sectors. Mengelkamp et al.[63] provides an energy trading approach between prosumers and consumers without the need for a middleman. The given approach is based on distributed ICT and a private blockchain. The results of a local energy market simulation between 100 residential households are discussed. The overview of the consensus mechanism in managing renewable energy certificates and smart grid operations was explained by Wang et al.[64].

Intelligent Transport System

Lei et al.[65] considered that secure key management methods were essential to keep networks safe. Therefore, they proposed a framework to capture the vehicle departure information integrated with blockchain-based network topology for transporting keys and then re-keying within the same security domain. The simulations and analysis demonstrate the effectiveness and efficiency of the proposed framework, indicating that the blockchain structure outperforms the traditional structure regarding key transfer time.

Table 2.1: Summary of Research Related to Usage of Blockchain in Various Application Areas

Ref. (s)	Application Area	Contribution(s)	Benefits(s)	Challenge(s)
[37,38]	Finance & Crypto-currency	Digital and cross-border payment, streamline banking and lending services	Improved security and transparency, reduced transaction costs, and counterparty risk	Scalability, regulatory hindrances, and high energy consumption
[42,43]	Supply Chain Management	Product traceability, inventory tracking, provenance verification, and anti-counterfeiting	Enhanced transparency and traceability, Reduced fraudulent activities	Interoperability, scalability, and data privacy issues
[47,48,49]	Healthcare	Managing EHR/ PHR, safeguarding genomics data, clinical trials, and preventing healthcare frauds	Data security and interoperability, enhanced data integrity and confidentiality	Scalability, Standardization, patient consent management
[56]	Voting Systems	E-voting	Secure and transparent elections	Voter secrecy, scalability
[57,58]	Education	Digital locker for student's credentials management and secure data sharing, issuing and verification of academic certificates	Transparency, credibility, accessibility, traceability, fraud resistance system	Management cost, regulation complexity, privacy, and scalability
[59,60]	Real Estate	Property transactions, Land registries, and escrow services	Fraud reduction, streamlined real estate transactions	Legal and regulatory barriers, integrating issues, and user adoption
[61,62]	Energy	Peer-to-peer energy trading, grid management	Enhanced efficiency, transparency, and reduced costs	Regulatory issues, technical integration, and market adoption

2.2 Storage and Querying Techniques for Blockchain Systems

This section describes the different storage and querying techniques for blockchain systems. The summary of the research related to storage and querying techniques is described in Table 2.2

Direct Querying

Direct Querying is the process of retrieving data placed on the blockchain by directly accessing the blockchain nodes. This technique leverages the inherent data structures of the blockchain and the APIs offered by blockchain platforms to retrieve the stored information. This method ensures robust data integrity and immutability as the data is securely kept on the blockchain, making it resistant to tampering. It is highly effective in simple use cases, such as retrieving transaction histories and accessing specific data blocks [66].

Off-Chain Querying

Off-chain querying involves retrieving data that is not directly stored on the blockchain but instead stored in the external databases or cloud storage, with only references on the blockchain [67,68]. Cryptographic hashes or other referencing mechanisms link the data back to the blockchain. It mainly manages large amounts of data, such as medical records, supply-chain records etc. Thus, it addresses the limitation of on-chain storage and reduces the load on the system, enhancing system performance and scalability. However, it suffers from challenges in preserving integrity and consistency between the on-chain and off-chain storage.

Smart Contract-Based Querying

Smart contract-based querying utilizes blockchain technology to automate and enforce data queries. This enhances security and trust by automating business

processes and data access policies[69]. However, creating or designing contracts may include inherent risks such as coding errors and security vulnerabilities. Executing these contracts consumes a lot of computing costs, especially for complex queries. Despite these obstacles, employing smart contract-based querying is very suitable for applications that include healthcare and supply-chain, where the secure handling of data access is crucial[70].

IPFS Integration Based Querying

IPFS is a decentralized, distributed, and content-addressing-based file storage system that is integrated with blockchain by various researchers to handle massive files efficiently. To provide data integrity, the cryptographic hashes are kept in the blockchain, while actual records are placed in the IPFS network. This approach combines the benefits of both systems, i.e. the immutability and security of blockchain with the scalable, decentralized storage facility of IPFS[71]. However, preserving consistency between the off-chain files and the on-chain references, as well as guaranteeing data availability and privacy, are some of the issues applications that need to store massive amounts of data, such as multimedia files, huge healthcare datasets, or lengthy papers. This approach can benefit significantly from IPFS integration while still taking advantage of blockchain's security characteristics[72].

Zero Knowledge Proofs

ZKP is an advanced cryptographic technique that protects privacy by preventing sensitive data from being revealed during searches and verifications[73]. These are quite helpful in blockchain applications where security and anonymity are crucial. These methods are computationally expensive and sophisticated. However, academics use this method to verify identities and demonstrate transaction corrections without disclosing actual information[74].

Table 2.2: Summary of Research Related to Storage and Querying Techniques

Ref.(s)	Technique	Benefits	Challenges	Use Case (s)
[66]	Direct querying	Data immutability and integrity	Scalability, and performance issues with large data sets	Immediate queries in e-commerce platforms to check product availability, and instant balance
[67,68]	Off-Chain querying	Improves scalability, reduces on-chain storage load	Trust issue	Large healthcare datasets, historical data queries
[69,70]	Smart contract based querying	Automated secure queries	Complex, high computational costs	Automated and conditional queries in applications such as insurance, voting systems
[75,72]	IPFS Integration Based	Decentralized, scalable storage	High latency and network congestion	Large multimedia files
[73,74]	Zero Knowledge Proofs (ZKP)	Privacy-Preserving secure querying	Computational overhead	Identity verification, Privacy sensitive queries

2.3 Summary of Research Related to Blockchain-Based Electronic Health Record (EHR) Systems

2.3.1 Blockchain Platforms and Network Types in EHR Systems

This section summarizes the work done by various researchers related to Blockchain-Based Electronic Health Record (EHR) systems, considering the blockchain platform utilized, the type of network deployed, and their benefits and limitations.

After the development of the Ethereum platform in 2015 by Vitalik Buterin, Azaria et al.[51] in 2016 proposed a decentralized EHR management system called MedRec using the Ethereum Platform. In their work, smart contracts are designed to provide data confidentiality, user authentication, and secure sharing of medical records. POW consensus protocol is used to aggregate medical data. In the case of Ethereum, the vulnerability assessment of the smart contracts is essential to identify the loopholes in the designed contracts for efficient working of the system with less gas consumption[76]. An integrated deep learning-based model is proposed by the authors[77] for vulnerability detection in SCs. Also, in their other work[78], they detect loopholes in SCs to reduce gas consumption. Blockchain Network utilized in[79,80] operates on POW consensus mechanism, known for its high energy consumption[81] and its associated performance challenges. In a similar work, Xia et al.[82] proposed a MedShare system where all transactions are stored in a tamper-proof manner while sharing data from one entity to another. The authors[83] proposed Bheem as a framework for analyzing the privacy and security issues in healthcare 4.0. Many existing systems also require cryptocurrency tokens to launch transactions for records uploading and retrieving.

To improve the interoperability among existing EHR databases, Yang and Li[84] proposed an architecture that is independent of any existing platform. It safeguards EHRs from alterations and malicious usage by monitoring the transactions taking place within the database. However, query handling with EHR management needs to be considered. Li et al.[85] introduced a blockchain-driven framework for the secure storage of medical data. The proposed framework ensures originality and verifiability of stored data while safeguarding the patient's privacy using cryptographic algorithms. The authors of[86] proposed a blockchain-driven solution for securing EHR systems. The proposed solution assures the integrity of data and interoperability among the existing systems. The authors of[87] proposed a framework called BinDaas using blockchain and deep learning technology. Blockchain provides security to users' EHR and deep learning

to predict future diseases. Most of the existing work focuses on the security and privacy of information and uses a relational database to handle unstructured medical data. EHR systems that have the ability to handle queries or access specific data from the chain on time are required.

Using the *Hyperledger Fabric (HLF)* platform, several researchers[88,89] proposed solutions for handling and sharing EHRs. The work focuses on providing solutions where the patient has control over their records. They used smart contracts to define access policies. Yang et al.[90] introduced architecture independent of any specific platform for securing existing EHR systems. The authors of[91] proposed a permissions-based EHR management system. This system provides a secure mechanism for sharing medical records along with handling queries. The CouchDB database is used to store data. However, with an increase in the size of the network, the overhead increases, affecting the system's performance. Thus, several researchers made significant efforts to secure, store, and query healthcare data, but still, some more focus needs to be considered for the efficient handling of such massive unstructured data. The authors of[92] discuss various factors that affect the performance of the network and analyze the performance considering parameters such as throughput and latency. The summary of the research work related to blockchain platforms and network types in EHR systems is shown in Table 2.3.

Table 2.3: Summary of Blockchain Platforms and Network Types in EHR Systems

Author(s)	**Year**	**Objective**	**Blockchain Platform**	**Network Type**	**Benefit(s)**	**Limitation(s)**
Azaria et al. [51]	2016	To provide a Medrec management scheme for handling EMRs.	Ethereum	Public	Provides authenticity, confidentiality, and secure data sharing.	Less energy efficient as POW consensus algorithm is used.
Xie et al. [82]	2017	To develop a data sharing system at outside premises, i.e. among cloud providers.	Not Mentioned	Private	Transactions are stored in a tamper-proof manner, providing an access control mechanism.	No use of NoSQL databases to handle unstructured data.
Sukhwani et al. [92]	2017	To design a permission-based blockchain platform and analyse the performance.	Hyperledger Fabric	Private	Shows the effect of parameters on performance.	Limited Scalability, no query handling.
Yang and Li [84]	2018	To design an architecture using relational databases for improving the interoperability of the existing EHR system.	Independent of any specific platform	Private	Independent of any specific platform and compatible with existing EHR system.	Simple queries being handled.

Li et al. [85]	2018	To introduce a model for maintaining patient privacy and secure storage of medical data.	Ethereum	Public	Effectively resist data tampering or deletion operations, Cryptographic Algorithms with memory management help to manage leaked data.	Limited focus on access control policies. Also, the structure by which blocks are stored needs to be optimized.
Yang et al. [90]	2019	To design a blockchain-based system for securing EHR systems.	Hyperledger Fabric	Private	The integrity of existing medical data and interoperability among existing EHR systems.	Sharing of EHRs not discussed.
Hang et al. [91]	2019	To design a blockchain-based novel EMR management platform.	Hyperledger Fabric	Private	Complete, immutable log and easy access to medical information.	Sharing of EMRs is limited to hospitals only and not with third-party alliances.
Tanwar et al. [86]	2020	To propose an efficient EHR sharing system using Blockchain with enhanced security and privacy.	Hyperledger Fabric	Private	Permission-based EHR system for sharing records.	Type of database is not mentioned.

2.3.2 Comprehensive Review of Blockchain-Enabled EHR Systems

This section thoroughly examines various studies and projects integrating blockchain technology with EHR systems. It includes critical information such as references, publication years, descriptions of the systems, the technologies and algorithms used, evaluation methods, as well as the benefits and limitations identified in each study. This comprehensive review aims to present perspectives on the present state of blockchain applications in EHRs, highlighting technological advancements and identifying potential areas for improvement.

Blockchain-based EHR systems leverage a range of modern technologies, including cryptographic algorithms, SCs and distributed storage, to enhance security, reliability, and interoperability. Encryption and Decryption techniques are employed by different authors along with blockchain technology for designing secure, reliable, decentralized, and transparent healthcare systems. The authors of[93] introduced SECURE-BLOCK, an innovative and resilient EMR sharing system using blockchain as a ledger, and improved cryptography using Rivest, Shamir, Adleman (RSA) and the Blowfish Algorithm. Blockchain technology, access control policies, and cryptographic methods are utilized to overcome the shortcomings of current systems and guarantee the privacy and confidentiality of EMRs.

The researchers of[94] introduced a secure record-keeping and sharing approach using blockchain technology to address the vulnerability of EHRs to cyber-attacks and data breaches. They presented a distributed off-chain model utilizing the IPFS to store massive amounts of medical data, enhancing scalability and ensuring content integrity. The system incorporates a Ciphertext Policy Attribute-Based Encryption (CP-ABE) algorithm with blockchain technology to enable precise access control so that only users with the necessary attributes can access specific EHR data.

Koufi et al.[95] examined the use of Hyperledger Fabric in healthcare, focusing on customized EMR implementation and blockchain technology. It highlights the open-source initiative's flexibility and its implementation challenges. The article suggested further research, including improving system design and establishing secure communication channels. Jing et al.[96] in their work highlights the benefit of using multichannel and also proposed a multi-channel blockchain-based model for asset trading.

Díaz and Kaschel[97] suggested a blockchain-powered EHR management system that aims to tackle scalability concerns and improve the privacy and confidentiality of medical data. In order to demonstrate its efficacy, the performance evaluation is conducted by examining the transaction throughput and latency of both read and write queries. The researchers[98] presented a novel access control scheme called Blockchain-based Access Control Scheme (BACS) with multi-party authority to enhance the security of transmitting EHR information. The system employed a Lightweight Fused Cryptographic method, patient and doctor signatures, and a SC to verify the transaction's validity. Further, the emergence of scalability issues while using blockchain as a storage solution for healthcare data is discussed in the literature[99]. The summary of the related blockchain-enabled EHR systems is discussed in Table 2.4.

Table 2.4: Comprehensive Review of Blockchain-Enabled EHR Systems

Author (s)	**Year**	**Objective**	**Technology**	**Algorithm**	**Evaluation**	**Benefits**	**Limitations**
Walzade and Vikhar [93]	2024	To introduce SECURE-BLOCK, an innovative and resilient EMR sharing system.	Java based blockchain	Hybrid RSA+ Blowfish algorithm	Encryption/ Decryption time w.r.t varying data size	Data integrity and confidentiality	Limited evaluation parameters considered.
Kaur et al. [94]	2023	To introduce blockchain and attribute-based access control method for healthcare data management.	Remix-Ethereum IDE	CP-ABE, Smart Contracts, IPFS	Security Analysis, Execution time for designed algorithms	Tamper- Proof, verifiable audit trail, scalability	Additional overhead while utilizing CP-ABE in real-time applications.
Vidhya and Kalaivani [98]	2023	To present a secure blockchain-based scheme to safeguard networks from both internal and external attacks during medical data sharing.	Not Mention	LFC, smart contracts, Proxy-reencryption oracles	Security and computation cost analysis	Data integrity, confidentiality and resistant to some attacks	Overhead associated with proxy oracles.
Ndzimakhwe et al. [100]	2023	Patient–Centric framework using hyperledger is proposed.	Hyperledger Fabric	Smart Contracts	Not discussed	User-friendly interface, patient centic and data confidentiality	The scalability issue needs to be addressed.

Díaz and Kaschel [97]	2023	To propose a scalable blockchain-based EHR management system to address medical data's security and privacy concerns.	Hyperledger Fabric	Smart contracts	Transaction throughput and latency	Scalability, Decentralization	Need to address EHR management of document storage in Hyperledger.
Sharma et al. [101]	2022	To introduce a Blockchain and CP-ABE based secure storage and access control along with revocation process.	Java-based blockchain network design	Cloud and CP-ABE	Time taken by key generation, encryption algorithms, etc.	Improved data access control and flexibility in revocation.	High computational cost and potential scalability issues.
Ali et al. [102]	2022	To propose a novel approach based on the neural network and group theory that efficiently detects invasions in the IoT network.	Hyperledger Fabric, Cloud	Homomorphic Encryption	Encryption/ Decryption time, latency, and throughput.	Enhanced security and searchability of encrypted data.	Limited evaluation in real-world scenarios and potential performance overhead.
Ali et al. [103]	2021	To propose lightweight deep learning-based privacy preservation model for IIoMT.	Ethereum , Cloud	ABE based access control	Analysis of access policies w.r.t attributes and certificate authority, etc.	Improved security through multi-certificate authorities.	Complexity in managing multiple certificate authorities.

Li et al. [104]	2021	To achieve secure storage, trustworthy sharing, access management, and privacy of healthcare records, an EHRchain named blockchain-based system is proposed.	Hyperledger Fabric, IPFS	Attribute-based and homomorphic encryption,	The time taken by various proposed algorithms are discussed.	Enhanced security and privacy through advanced cryptographic techniques.	Computational overhead due to homomorphic encryption.
Ali et al. [105]	2021	To achieve security & privacy in patient healthcare networks with higher efficiency at lower cost, a blockchain based framework is proposed.	Hyperledger, Ethereum and Cloud	Ring Signature + ABE Smart Contracts	Execution time	Enhanced overall security, privacy, and reliability of EHRs.	Potential scalability issues and high resource consumption.
Mubarakali [106]	2020	To monitor the healthcare services in Cloud and securely transfer the data from patients to Cloud Storage by the proposed model named SRHB (Secure Robust Healthcare Blockchain).	Implemented using NetBeans 8.2 and Jelastic cloud environment	ABE-based access control	Success Rate, Average Delay, and execution time.	Improved security and robustness for cloud-based healthcare services.	Limited real-world testing and potential scalability issues.

Saravanan and Umamakeswari [107]	2020	To preserve the privacy of PHR data, Hashed based access authentication scheme is proposed	Cloud storage	Hashed Based CP-ABE	Encryption/ Decryption Time	Enhanced privacy and security for personal health records.	Limited by the complexity of the encryption scheme and potential performance overhead.
Shahnaz et al. [108]	2019	To design a system using Blockchain Technology for EHR and provide secure storage to EHRs.	Ethereum, IPFS	Role Based Smart Contracts are implemented	Execution time, Throughput, and Latency.	Improved data integrity and access control.	Scalability issues and high computational cost.
Thwin and Vasupongayya [109]	2019	To present a blockchain model for preserving the privacy of PHR data.	Hyperledger Fabric, Cloud	Proxy re-encryption and Access Control List	Time taken by various cryptographic algorithms considering the varying data size and users.	Strong privacy preservation and access control.	Focused on theoretical models with limited real-world testing.
Vora et al. [83]	2018	To provide efficient storage and maintenance of EHRs using blockchain.	Ethereum, Offchain storage	Smart Contracts	Not Mentioned	Enhanced security and immutability of health records.	Limited evaluation and practical implementation details.

It can be concluded from the above literature review that the current techniques have several shortcomings. Healthcare is a comprehensive sector involving numerous entities and health data is a valuable asset for patients. Therefore, such data must be securely stored and shared so that unauthorized users cannot perform malicious acts on it, as this data is highly susceptible to risk of breach. However, various existing systems do not consider the various healthcare stakeholders when developing models, neglecting patients as data owners who need to know exactly when, where and what data is shared and who can share and receive a complete medical history for timely treatment while maintaining privacy. Also, healthcare records are scattered at multiple hospitals and are not integrated. Thus, sharing them outside institutes becomes difficult.

In conclusion, despite much research on blockchain-based EHR systems, a permission-based framework with a decentralized, tamper-proof and immutable healthcare network is needed to solve the above problems.

2.4 Chapter Summary

This chapter discusses an overview of application areas that utilize blockchain technology, including different storage and querying techniques for blockchain systems. It also discusses relevant literature related to blockchain-based EHR systems. In addition, it also provides an in-depth study of existing blockchain platforms for healthcare data, highlighting the potential advantages and disadvantages. Based on the literature review, the next chapter identifies research gaps in existing technologies. Based on these research gaps, research objectives are then formulated.

Chapter 3

PROBLEM FORMULATION

This chapter presents the research gaps identified through an extensive literature review. The identified gaps highlight the challenges in the existing healthcare sector in managing health information using Electronic Health Record (EHR) systems. Based on these limitations and shortcomings, three research objectives have been outlined, focusing on developing a blockchain-based framework for securing and retrieving healthcare data. These three objectives aim to address the identified gaps, which will be systematically addressed in the upcoming chapters of the dissertation.

3.1 Research Gaps

Based on the extensive literature review, there were numerous research gaps, which were identified as follows:

1. EHR systems are mainly *centralized* as they are managed and controlled by hospitals. There are strict regulations and procedures for transferring sensitive data, such as medical records, outside the institute. Accessing fragmented records in case of urgency is not convenient. Moreover, a person has to undergo the same medical test that was previously performed at another hospital if medical history is not maintained.

2. There are issues in obtaining a *unified view* of health data as data is scattered at multiple places and is not integrated.

3. In EHR systems, the foremost priority is the *security and privacy of patient information*. Patient information must be placed securely so that unauthorized users cannot perform malicious acts. Although the existing EHR/EMR systems try to solve this problem using various cryptographic methods, they still cannot guarantee the complete security of the database.

4. The information in distributed applications requires a secure framework to store, share, and retrieve. Due to the lack of secure methods, information is prone to unwanted disclosure, security threats, or irrevocable loss.

5. Today, medical data is proliferating daily and is heterogeneous by nature. The heterogeneity of medical data must be considered when designing the architecture.

6. There is a need for efficient handling of data, such as images, PDFs, text, etc., coming from different data sources.

7. There are issues in *data interoperability* from various sources due to different syntactic attributes and formats.

8. Issues related to *traceability and non-repudiation* need to be considered.

9. *Scalability* is another concern related to the EMR system. The number of patients is rising at an exponential pace. The conventional EMR-based system is unable to effectively address the aforementioned situation. It is necessary to establish a scalable framework to address the aforementioned issue.

10. There is a requirement to develop and maintain a *user-driven* data integration framework, taking into account user preferences for data extraction and ensuring the system's privacy.

3.2 Research Objectives

The following are the objectives of the proposed research :

1. To study and explore various techniques and tools available for creating and managing Blockchain.

2. To propose and implement Blockchain based framework for securing and querying Healthcare data.

3. Testing and validation of the proposed framework using real-world data.

3.3 Research Methodology

The following methodology is followed to address the above-stated objectives. Figure 3.1 shows the flow of the research methodology followed.

Research Methodology for Objective 1:
A detailed study of existing tools and techniques for handling healthcare data using Blockchain is conducted. Furthermore, the framework proposed by various researchers is studied and explored to gain deep insights and a better understanding of the subject matter. The bibliographic study considers the following aspects of ensuring the security and querying of healthcare data.

- Understand the basic concept of Blockchain as a distributed ledger, including its characteristics, applications, and diverse set of available tools.

- Gained knowledge about the current technologies and methodologies employed to safeguard the security and privacy of healthcare data.

- Reviewed the existing work to understand the different types of healthcare data generated and their storage in various databases for the purpose of storage, query, and rapid retrieval.

Research Methodology for Objective 2:

To accomplish objective 2, a framework is designed and implemented in three successive stages:

- In the first stage, the framework architecture is designed by selecting a Hyperledger Fabric blockchain platform involving multiple healthcare sector entities. Then, data models for various entities and access control are defined to store EHR data on the designed blockchain network securely. A set of algorithms is designed and implemented using smart contracts defining the functionality of the patients, doctors, and Third-Party Administrators (TPA).

- CouchDB is incorporated as a state database to facilitate handling rich and complex queries considering both keys and data values, not only keys. In addition, the scalability of the proposed framework is enhanced by the integration of distributed storage IPFS, which stores massive amounts of data off-chain and their corresponding hash values on the Blockchain.

- In the second stage, IB-PRE algorithm is designed and implemented to improve privacy preservation and efficient management of EHRs. The implemented IB-PRE algorithm ensures the secure sharing of records while preserving privacy by preventing the disclosure of private keys.

- The third stage improved the proposed framework performance, security, and query capabilities by implementing a dual-channel architecture combined with two robust cryptographic algorithms, RSA and AES. The proposed work includes the features of private data collection and ACL to facilitate confidential and secure communication.

Research Methodology for Objective 3:

- The proposed framework is tested and validated by conducting a qualitative assessment of the designed contracts to ensure proper functionality and

interaction among various components. The system performance is evaluated by considering different scenarios, including varied transaction loads, transaction types, number of channels, etc.

- The security analysis of the proposed framework is also conducted through two vulnerability assessment tools, namely, Chaincode Analyzer and Oyente.
- In addition, the effectiveness and efficiency of the proposed framework are evaluated and compared with existing related works. The evaluation metrics include security attack resistance, features offered, scalability, transaction throughput, and latency.

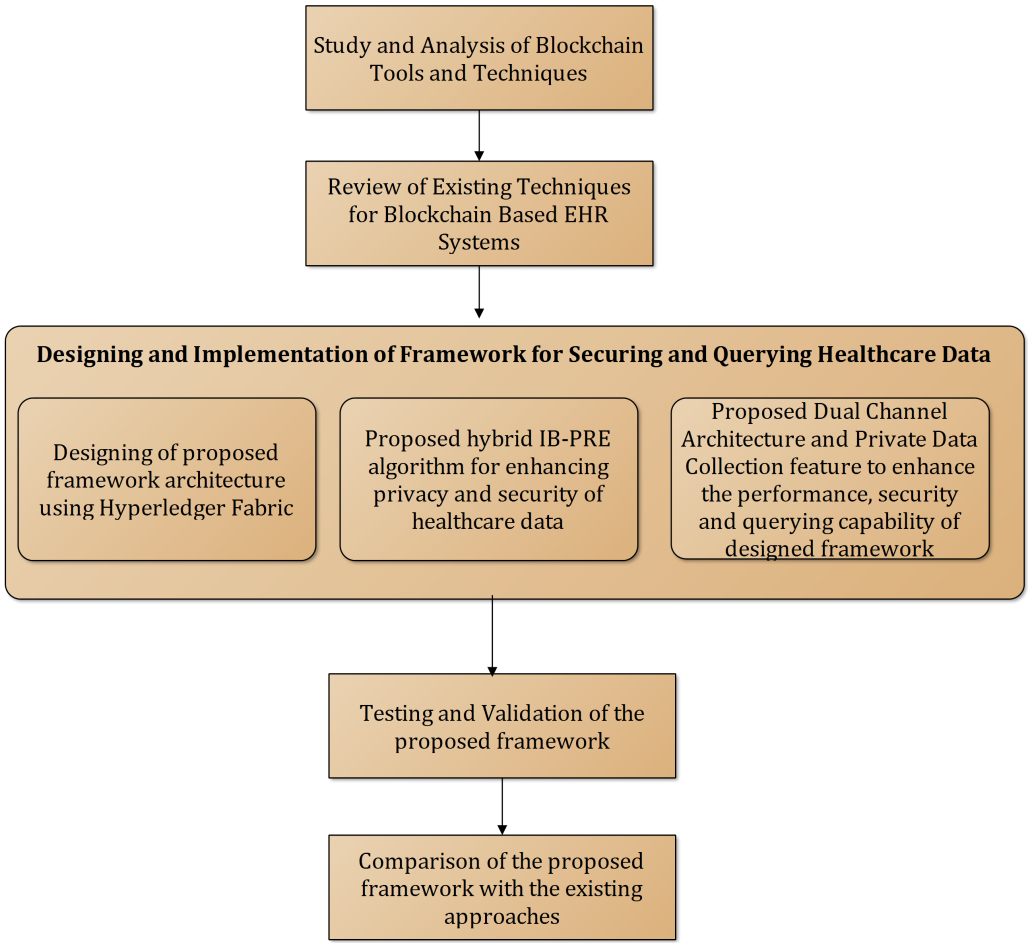


Figure 3.1: Research Methodology

3.4 Chapter Summary

This chapter discusses numerous research gaps based on the extensive literature review. Three objectives are formulated to address some gaps in the existing research. Further, to address the objectives, a research methodology is designed and discussed in the last section of this chapter. In the next chapter, a permissioned blockchain-based architecture encompassing multiple identities in healthcare is proposed and discussed in detail.

Chapter 4

BLOCKCHAIN BASED FRAMEWORK FOR SECURE STORAGE, SHARING AND QUERYING OF ELECTRONIC HEALTHCARE RECORDS (EHRs)

This chapter aims to design a permissioned Blockchain-based framework using HLF to provide secure storage, sharing, and querying of EHRs. Algorithms have been designed to define the functionality of patients, doctors, and TPA using smart contracts. In the existing approaches, LevelDB (Key store database) is used as a state database that is not suitable for handling complex queries. To overcome the above issues, CouchDB (document store database) is utilized as a state database, where the search is based on keys and data values rather than only keys. To measure the various performance indicators for this framework, a benchmarking tool *"Hyperledger Caliper*" is used. The findings of the experimental evaluation of the proposed framework demonstrate its effectiveness and efficacy.

4.1 Motivation

EHR contain vital, critical, and confidential information of any person. These records are susceptible to various security risks, including the loss of data,

unauthorized data disclosure, malicious alterations, and other concerns[110]. Thus, there arises a need for a decentralized, tamper-proof, and transparent healthcare network. The majority of current research centres around decentralized apps (DApp), smart contracts, and permission-less networks when discussing blockchain-based healthcare systems. They use a public blockchain-based platform, i.e. Ethereum, to securely access and share EHRs. However, EHRs contain the patient's confidential data that needs to be frequently stored, queried, and shared among various entities in a secure manner. However, in the permissionless network, the entities are anonymous, which makes the system untrusted. Hence, a permission-based framework with a decentralized, tamper-proof, and immutable healthcare network is needed to tackle the above issues.

In our work, the HLF platform is used for designing the proposed framework because of the following benefits:

i) *Computation and Operational Cost:* HLF platform does not require a native cryptocurrency like other platforms such as Ethereum for the mining process and smart contract execution. Significant attacks/risks can be reduced by avoiding cryptocurrency. A lot of power is consumed during the cryptographic mining process, which is not the case with the fabric platform. The absence of these operations reduces the extra cost involved and improves system performance significantly. Thus the system can be deployed with lesser computational as well as operational cost than any other distributed system.

ii) *Privacy and confidentiality:* In the fabric, every participant can be identified, and the network is permissioned, leading to enhanced privacy and confidentiality.

iii) *Scalability and Performance:* The HLF platform provides high throughput

and low latency compared to any public blockchain. The healthcare domain frequently shares high volume and fast-speed transactions, making it difficult for public blockchain to handle.

The following contributions are made to address the above issues.

- A secure and effective blockchain system developed for managing electronic healthcare records of all members in the healthcare industry.
- Algorithms have been designed to describe the functionality of patients, doctors, and TPA using smart contracts in *Hyperledger-Fabric*.
- In the existing approaches, *LevelDB* (Key store database) is used as a state storage database that is not suitable for handling complex queries, whereas in the proposed framework, *CouchDB* (NoSQL document store database) is used as a state storage database, allowing for searches based on both keys and data values rather than just keys.
- Experimental evaluation of the proposed framework is performed using the tool *"Hyperledger Caliper"* by considering different scenarios. In the existing work(s), the authors only consider a limited network size of 1Org with 1Peer, whereas in the proposed work, we have evaluated our system by considering different sizes such as 1Org with 1Peer, 2Orgs with 1-1 peer each, and 2Orgs with 2-2 peer each.

4.2 Tools/Platforms used to Design and Implement Proposed Framework

4.2.1 Hyperledger Fabric

HLF is an open-source, private, and permissioned platform hosted by the Linux Foundation. Due to permissioned property, each transaction is authenticated,

approved, validated, and tracked. In HLF, different roles are defined in terms of clients, peers, and ordering services. In the proposed framework, solo ordering is used as an ordering service.

- **Clients:** These are the applications that propose transactions on the network on behalf of users.
- **Peers:** Peers in the networks perform two main tasks i.e. to maintain the state of the network and to copy the ledger. Peers are categorized into two categories, i.e. endorsing peers and committing peers.
- **Ordering service:** The ordering service plays a vital role in reaching a common agreement and defines the order into which transactions are committed to the ledger. The ordering service provides 3 types of orders by which transactions are committed to the ledger such as Solo, Kafka, and Raft.
    1. **Solo:** This type of ordering is used by most of the developers that involves a single ordering node.
    2. **Kafka:** This type of ordering is used for handling real-time data where streamlined processing is required.
    3. **Raft:** It is a crash fault-tolerant mechanism that follows Raft protocol, i.e. "leaders and followers".

The working and step-by-step flow of the transaction in HLF[111] is explained below and diagrammatically shown in Figure 4.1.

1. The process starts after the client submitting a transaction proposal to endorsing peers.
2. Then endorsing peers by executing chain code simulates the submitted proposal and prepares endorsements.

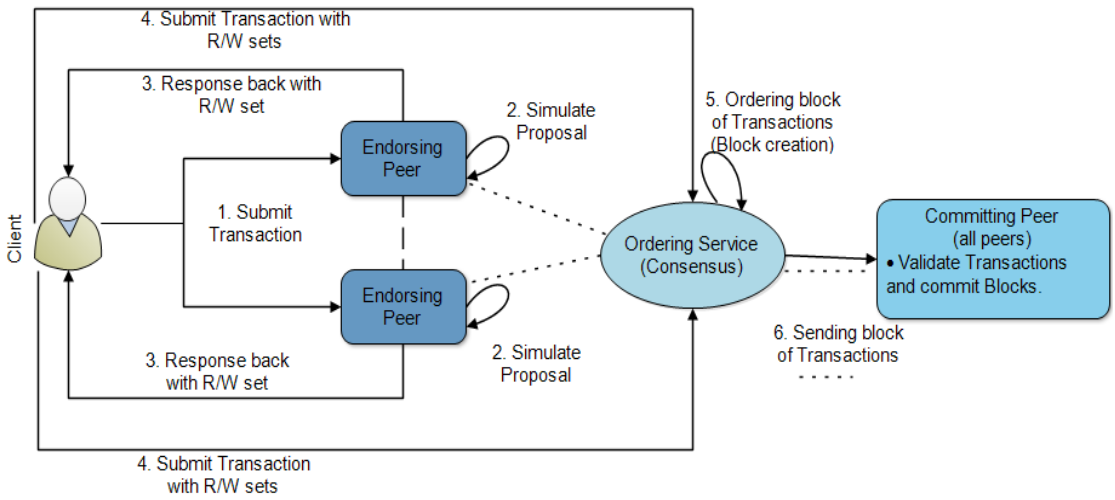


Figure 4.1: HLF Workflow.

3. After preparing endorsements, the peer's response back to the application client.

4. Client application after combining transaction proposal with endorsements broadcasts to orderers for ordering service.

5. Then, the ordering service checks the endorsements and creates blocks of transactions before broadcasting to all peers called Committers.

6. Then all peers, after receiving transaction blocks, validate each transaction, endorsement policy, commit block to the blockchain, and valid transactions to the state database.

4.2.2 HLF Ledger

HLF ledger consists of a blockchain, which serves as a series of linked blocks that maintain a permanent, ordered record, in addition to a state database that holds the current state information. The current state represents the latest and already existing values for all the keys, including those in the chain transaction log. The state database represents an indexed perspective of the transaction log of the chain, which can be reconstructed from the chain at any moment. HLF

platform provides two options for the state database. One is LevelDB, and the other is CouchDB. LevelDB is the default state database that stores chain-code data as key-value pairs. CouchDB is another choice that offers extensive query capabilities when the data of a smart contract is represented in JavaScript Object Notation (JSON) for conducting queries based on content. The diagram depicted in Figure 4.2 shows ledger states for one record, Record1 in CouchDB, comprising a key and corresponding value. To handle complex queries, CouchDB (document store database) is used for the proposed framework.

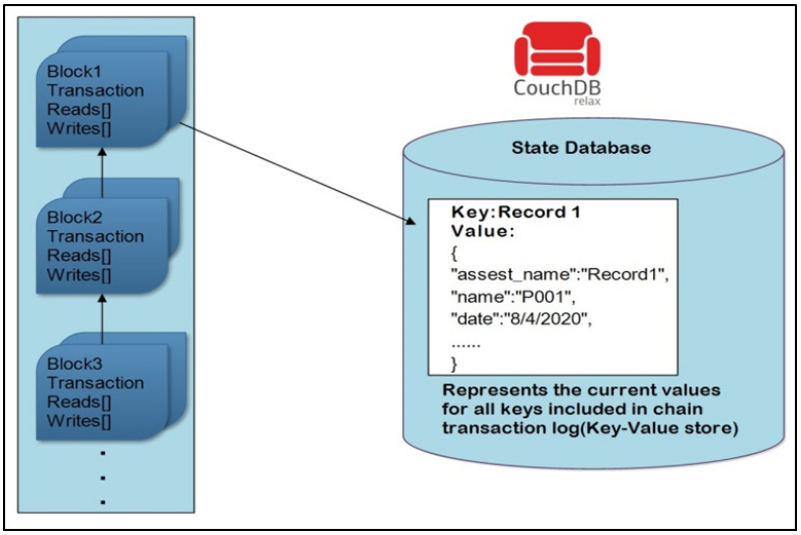


Figure 4.2: Distributed Ledger Overview

4.2.3 InterPlanetary File System (IPFS)

An IPFS system maintains different types of data, such as images, videos, and documents, across a distributed network. This system employs content-based addressing rather than location-based addressing [112]. Every file uploaded to IPFS is assigned a unique hash value or content identifier (CID), which can be used to retrieve the file at a later time. Hash or CID is a cryptographic number calculated from the contents of a file. When a file is uploaded in IPFS, it is kept as an IPFS object, which is divided into two parts: data and link. If the file's size is less than or equal to 256 KiloBytes, it is saved in a single object containing only a single data part and a null value in the link part. But suppose the file size exceeds

256 KiloBytes. In that case, it is automatically split into further multiple different objects, each of which can have up to 256 KiloBytes of data and/or links to other various objects. Each object calculates its own unique hash. Hashes of each object are represented in the form of Merkle Directed Acyclic Graph (Merkle DAG), which aggregates all objects hashes and gives a single root hash value that describes the entire file. The root hash value is used to identify the whole file and reconstruct the complete file from its split objects.

Along with Merkle DAG, it also maintains a hash table consisting of keys and values distributed among all network peers called Distributed Hash Table (DHT). Keys denote the location of the peer where the object is stored. Later, when the user wants to retrieve the data using hash, IPFS utilizes DHT to respond to the corresponding query.

The security of the proposed system is enhanced by keeping the encrypted medical records within IPFS while their hash values are recorded on the blockchain. Access to the confidential documents is restricted to authorized users only. Maintaining a complete record on the blockchain is an expensive process that affects the system's performance in terms of latency, response time, and scalability. The integration of hash values within the blockchain, along with the storage of the actual files on IPFS, makes the system more productive.

4.3 Proposed Framework

In this section, first, the architecture of the proposed framework for storing, sharing, and querying of EHRs is discussed. Second, the workflow of the proposed architecture is explained.

4.3.1 Architecture of the Proposed Framework

The architecture of the proposed framework includes three main phases, which are described as follows:

Phase 1: Authorization of Entities- The first phase is about authorization requests and responses. In this phase, participants such as doctors, patients, TPA, and admin request for registration to CA via client application or SDK using MSP. CA, in response, issues the certificate along with public/private keys to all users.

Phase 2: Storing EHRs- The second phase of the architecture is about patient monitoring, creating, and storing medical records. In this phase, the doctors first monitor the patient and generate records that are then placed in the blockchain by invoking smart contract functions. For scalability, the transaction information, along with the hash of the medical records, is added to the blockchain while actual records are kept in distributed storage, i.e. IPFS.

Phase 3: Sharing and Querying- The third and last phase of the architecture is the secure sharing and querying of medical records. In this phase, the patients grant/revoke access to doctors or TPA in response to requests for medical history access. Figure 4.3 describes the three phases of the architecture.

4.3.2 Workflow of the Proposed Architecture and Designing of Smart Contracts

The workflow of the proposed architecture is described in this section. It consists of four main components, i.e. Entities, Smart Contracts/Chain Code, Blockchain Network (BN), and Storage, as shown in Figure 4.3.

a) Entities: There are various entities involved in the proposed framework, such as admin, doctors, patients, TPA, and other stakeholders. Each entity has a different role and can perform multiple tasks by invoking the smart contract functions. The functioning of the patient, doctor, and TPA entity is explained in Algorithms 1 to 3, respectively. Table 4.1 lists all the abbreviations used for entities in the algorithms.

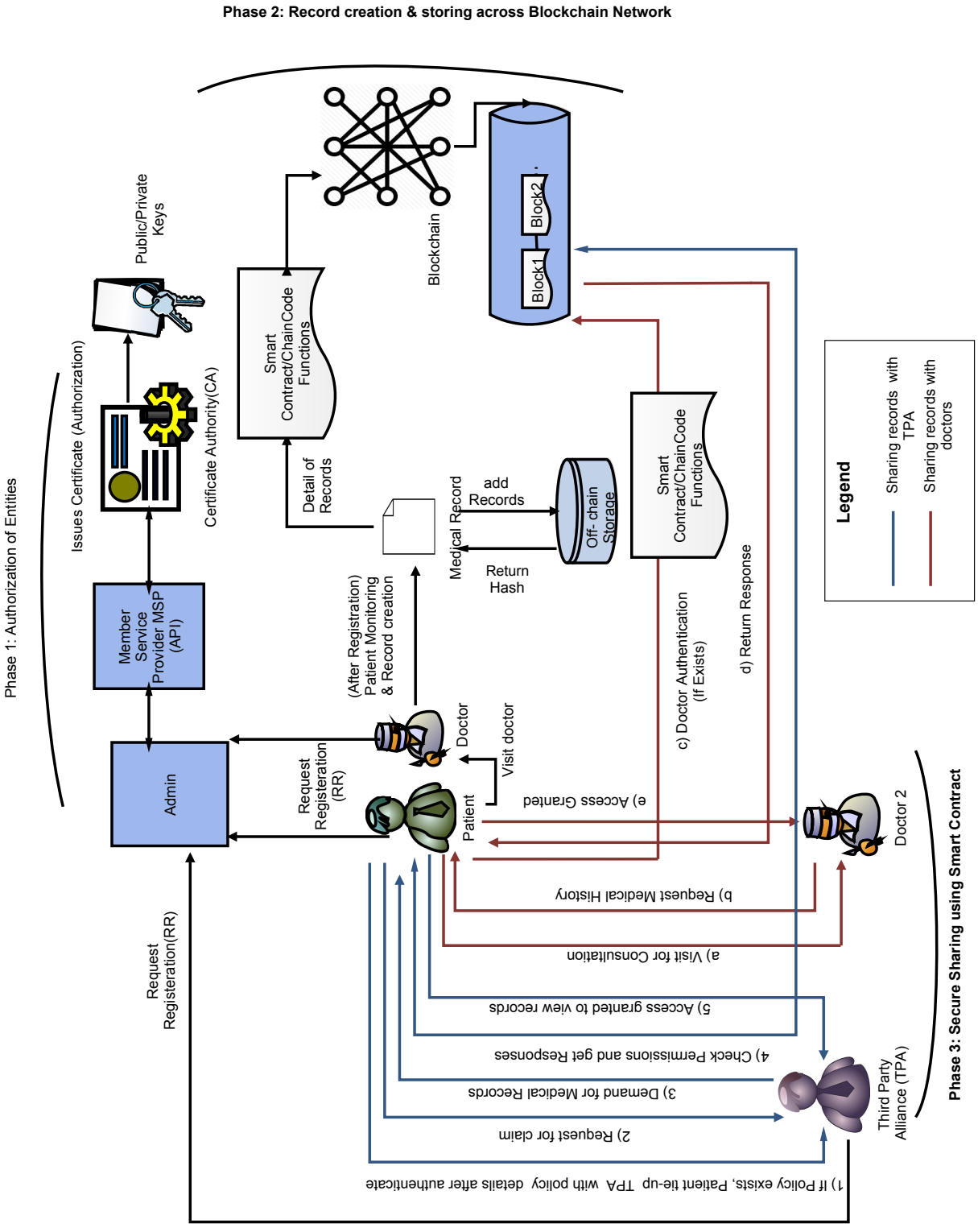


Figure 4.3: Architecture of the Proposed Framework

The functionality of the patient entity is given in Algorithm 1. In this, the patient requests for PID to system Admin for getting access to perform various transactions on BN such as read, write, query, and grant/revoke EHR records. First, by passing PID as an argument it checks if the patient already exists in the system. If it exists,

Table 4.1: Abbreviations used for Entities

PID	PatientID
DID	DoctorID
TID	Third Party Alliance (TPA) ID
BN	Blockchain Network
pEmail	Patient Email id

then the patient can handle various queries such as viewing his medical records, query them by date, and DID. Next, in case of consultation with other doctors, he first checks if the doctor exists and belongs to BN. Then grants access to the doctor to view and add records during treatment if the doctor exists. Otherwise, revokes access rights at any time from the doctor after consultation or treatment. Next, in case of having any medical policy, and for claim processing, the patient first checks if the TPA exists and belongs to BN. If it exists, then the patient will tie up with TPA with their policy details, including policy number, policy start date, policy valid up to, and other details. Also, grants access to the TPA to view records during claim processing. Otherwise, revoke access permissions at any time from TPA.

The functionality of the doctor entity is given in Algorithm 2. In this, the doctor requests for DID to system Admin for getting access to perform various transactions and handle queries on BN such as viewing registered patients and reading and writing EHR records of patients. First, by passing DID as an argument it checks if the doctor already exists in the system. If it exists, then the doctor can query from the BN, such as viewing medical records of already treated patients and query them by date and PID. Otherwise, he/she has to first register themselves with basic information. Next, if the doctor wants to access the patient's complete medical history, he/she sends the request to the patient to access the EHRs. The patients can grant or revoke the access rights to the doctor. If permission is granted, then the doctor will be able to access the patient's comprehensive and accurate medical history along with adding records during treatment.

The functionality of the TPA entity is given in Algorithm 3. In this, the TPA requests

Algorithm 1 Functionality of Patient Entity

Input: PID, DID, pEmail
Output: Get access to patient transactions

```
1: procedure PATIENT(PID)
2:     while true do
3:         if PIDexists then                          ▷ checks patient exists in BN
4:             print patient already registered       ▷ display message patient exists
5:             viewRecords (PID, pEmail)
6:             queryRecordByDate (PID, Date)
7:             queryRecordsByDID (PID, DID)
8:             queryRecordsByTID (PID, TID)
9:         else
10:            print "patient not exists, first register through Admin"
11:        end if
12:        if visitForConsultation then
13:            if Doctorexists then                   ▷ checks doctor exists in BN
14:                checkPermissions    ▷ access permissions such as view,write etc.
15:                if !allowed then
16:                    grantPermission (PID,DID)
17:                else
18:                    revokePermissionAfterConsultation(PID,DID)
19:                end if
20:            else
21:                Doctor not exists              ▷ display message doctor not exists
22:            end if
23:        end if
24:        if IsPolicy then          ▷ if patient has any medical insurance policy
25:            if TPAexists then                      ▷ checks TPA exists in BN
26:                patient tie-up with TPA with policy details
27:                checkPermissions
28:                if !allowed then
29:                    grantPermissionForRecordsAccess()
30:                else
31:                    revokePermissionAfterClaimProcessing()
32:                end if
33:            else
34:                TPA not exists
35:            end if
36:        end if
37:    end while
38: end procedure
```

for TID to system Admin for getting access to perform various transactions and handle queries on BN such as viewing registered patients list and EHR records of patients for the claiming process. First, by passing TID as an argument it

Algorithm 2 Functionality of Doctor Entity

Input: DID
Output: Get access to doctor transactions

```
1: procedure DOCTOR(DID)
2:     while true do
3:         if DIDexists then
4:             print Doctor already registered
5:             viewPatientRecord(DID,PID)  ▷ patients already treated or patients that granted access
6:             queryRecord(DID,PID)
7:         else
8:             print doctor not exists
9:             registerDoctor(basic Info)
10:        end if
11:        if grantedAccess then                    ▷ checks for access granted
12:            addRecords(DID,PID,Record) ▷ if true then add and view records of patient with PID
13:            viewPatientHistory(PID)
14:        else
15:            requestDataAccess() ▷ if not granted access then, request for access
16:            viewAlreadytreatedPatients()
17:        end if
18:    end while
19: end procedure
```

checks if the TPA already exists in the system. If it exists, then TPA can query from the BN, such as viewing medical records of already registered patients for the claiming process and query them by date and TID. Otherwise, he/she has to register himself/herself with basic information. Next, in case if TPA wants to access the patient's medical history then he sends the request to the patient for accessing the EHRs. The patients can grant or revoke access rights to TPA. If granted access then TPA can process the medical claim of the patient without delay.

b) Smart Contracts: In HLF, smart contracts are also referred to as chain codes. In chain code, the application logic is written using Go language [113], which runs on the top of a blockchain. In a smart contract, structures of different entities, such as doctors, patients, TPA etc, policy, and medical records, are described as shown in Figure 4.4. In the Go language, structure (struct) is a typed collection of different fields which group to form a record.

Algorithm 3 Functionality of TPA Entity

Input: TID
Output: Get access to TPA transactions

```
1: procedure TPA(TID)
2:     while true do
3:         if TIDexists then
4:             print TPA already registered  ▷ prints message TPA already exists in the BN
5:             viewPatientRecord(TID,PID)
6:             queryRecord(TID,PID)
7:         else
8:             print TPA not exists
9:             registerTPA(basic Info)
10:        end if
11:        if grantedAccess then                    ▷ checks for access granted
12:            viewPatientHistory(PID)  ▷ view patient record by entering patient id (PID)
13:        else
14:            requestDataAccess()              ▷ TPA view patients already tie-up
15:            viewAlreadyClaimedPatients()
16:        end if
17:    end while
18: end procedure
```

The doctor struct is defined in Figure 4.4 (a) consists of the following fields, each of which are string data type:

- DID: DID is the unique doctor ID.
- Name: Name field defines the name of the doctor.
- Specialization: It is the field that defines the specialization area of the doctor like Cardiologist, Surgery, etc.
- DContactNo: DContactNo is the contact number of the doctor.
- PatientList: PatientList is a field that comprises records of patient IDs linked to the doctor regarding treatment or consultation services.

Figure 4.4 (b) represents the struct type phealth. EHR structure which contains the below fields:

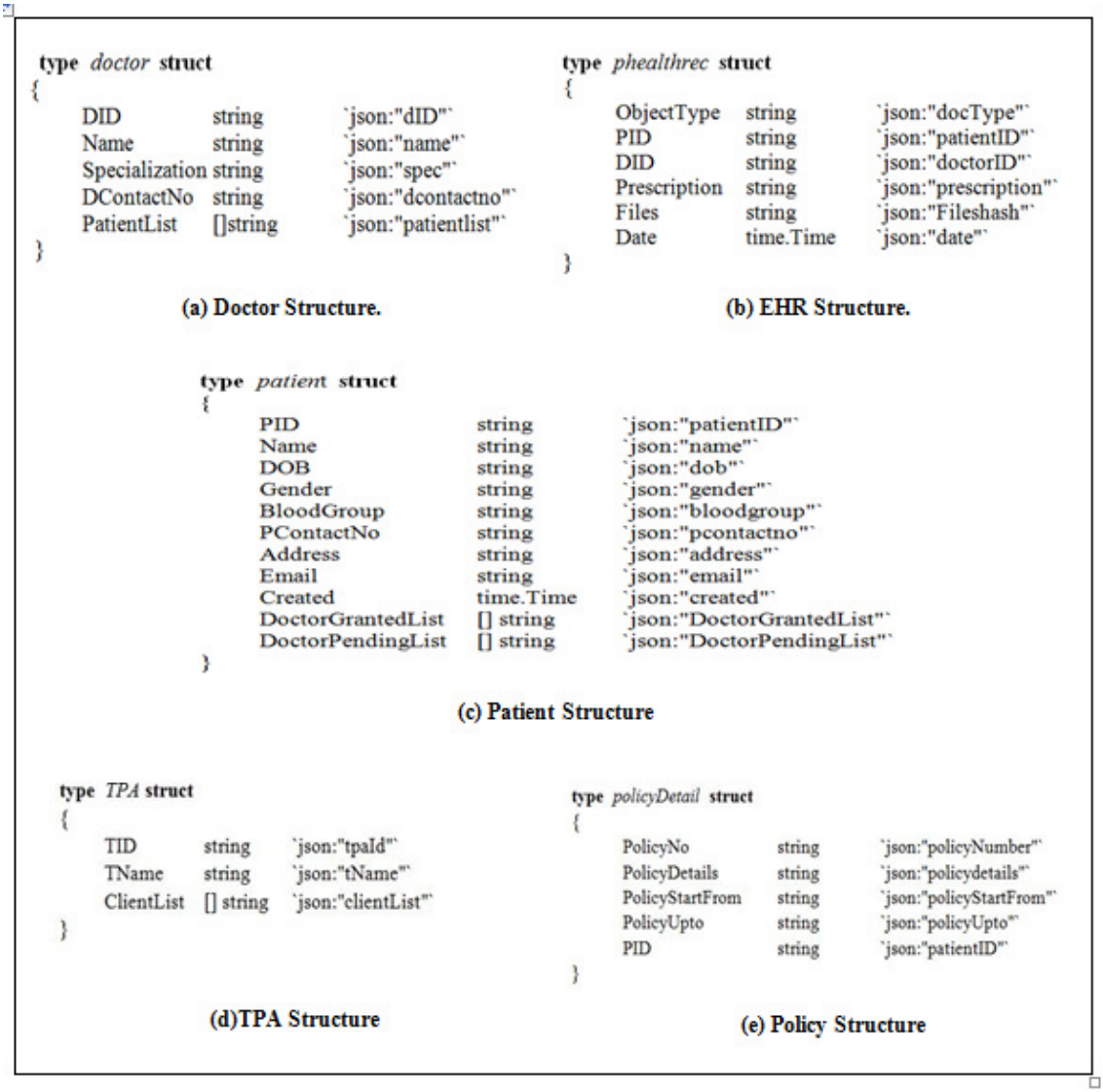


Figure 4.4: Structure of Various Entities and Medical Records

- ObjectType: ObjectType field is used in a database to distinguish different types of objects.
- PID: PID is the unique patient id that is assigned during the registration process.
- DID: It is the unique doctor id as already defined in doctor structure.
- Prescription: This field contains prescription medicines to patients by doctors. Many values can be inserted and separated by a comma. Comments and descriptions can also be provided in lab test reports and X-rays.

- Files: This field includes the hash value of records such as lab tests, X-rays, MRI images, etc. These are described in the prescription field.
- Date: Date field is of time data type. This field documents the timestamp indicating when the medical record is generated and preserved within a blockchain system.

Figure 4.4 (c) shows the patient structure, which consists of the following fields:

- PID: It is a unique patient id.
- Name: Name field stores the name of the patient.
- DOB: DOB stands for date of birth of the patient.
- Gender: Gender defines the gender of the patient.
- BloodGroup: defines the Blood Group of the Patient.
- PContactNo: This field defines the contact number of the patient.
- Address: The complete address of the patient is recorded in this field.
- Email: The email field is the field where the patient's email id is recorded.
- Created: The time at which the patient is registered with his basic information.
- DoctorGrantedList: This is the field that is of a list type. This contains the list of all the doctors to whom access is granted for medical records.
- DoctorPendingList: This is the list of pending doctor requests.

Figure 4.4 (d) represents the TPA structure which consists of the following fields:

- TID: It is a unique TPA id assigned during the registration process.

- TName: This field contains the name of the TPA.
- ClientList: This is the list of the patients who are associated with that particular TPA and granted access to view medical records.

Figure 4.4 (e) represents the policy structure which consists of the following fields:

- PolicyNo: PolicyNo is the number of the medical policy hold by PID.
- PolicyDetail: This contains the complete details of the policy such as policy name, term, coverage area, the sum assured, etc.
- PolicyStartFrom: It represents the effective date and time when the policy officially becomes active.
- PolicyUpto: It denotes the date and time when policy coverage ends.
- PID: This is the patient id to which the policy is associated.

Application logic is defined in the form of functions that are invoked through invoking a function of chain code. The invoke function of chaincode invokes different functions triggered by various entities in the network, which is represented in Figure 4.5. The proposed model executes various functions which are included in the functionalities of patients, doctors, and TPA algorithms. The following are the functions:

i) Viewing complete medical records EHRs by patients.

ii) Querying records based on DID/TID and date.

iii) Requesting patients for data access.

iv) Granting permission to other doctors/TPAs for accessing the record and revoking after permission.

The constraints for accessing which functions by whom and under what conditions

```
func (s *SmartContract) Invoke (stub shim.ChaincodeStubInterface) sc.Response {
        function, args: =stub.GetFunctionAndParameters ()
        fmt.Println ("invoke is running " +function)
        if function ="initLedger" { //handles different functions
                return s.initLedger (stub)
        }
        else if function ="registerPatient" {
                return sregisterPatient (stub, args)
        }
        else if function ="addRecord" {
                return s.addRecord (stub, args)
        }
        else if function ="viewRecords" {
                return s.viewRecords (stub, args)
        }
        else if function ="queryRecordsByDID" {
                return s.queryRecordsByDID (stub, args)
        }
        else if function ="grantPermission" {
                return s.grantPermission (stub, args)
        }
         else if function ="revoke" {
                return srevoke (stub, args)
        }// other functions
}
```

Figure 4.5: Snapshot of Smart Contract Functions

with access permissions are all enforced through chain code functions. The parameters used in algorithms are already declared in the above structures. Pseudocodes for the above four functions are presented in the form of Algorithms 4 to 8, respectively.

i) View Records by patient corresponding to PID: The records of the patients can be viewed from anywhere at any time by invoking *viewRecords( )* function shown in Algorithm 4. This function takes two arguments as input, where argument 1 corresponds to PID and argument 2 to email. Both values are then checked. If true, then only return the records corresponding to PID. Else, print an error message. The records are fetched from the state database (ledger), which is CouchDB.

ii) Query records by DID: *queryrecordsByDID( )* function explained in Algorithm 5 is similar to the *viewRecords( )* function as here the medical records are fetched based on the DID.

Algorithm 4 *viewRecords ( )*

Input: args1, arg2
Output: fetch all medical records (EHRs) corresponding to key entered as arg1

1: **if** $PID == arg1 \&\& Email == arg2$ **then** ▷ if both matched
2: return(records from the ledger) ▷ then return all records as described in phealhrec struct
3: **else**
4: print Invalid PID and Email ▷ else print message Invalid PID and Email
5: **end if**

Algorithm 5 *queryRecordsByDID ( )*

Input: PID, Email , DID
Output: fetch all records containing particular DID

1: **if** $PID \&\& Email$ **then** ▷ if both matched
2: Enter DID
3: return (all records containing DID)
4: **end if**

iii) Doctor requesting access to records during consultation or treatment: During a consultation if a doctor wants to access records to view medical history, he first requests the patient for access. Algorithm 6 is invoked by the doctor with his id "DID". The patient first checks the authenticity of the Doctor i.e. if the doctor exists on the network. If true, then the patient searches in access granted list (DoctorGrantedList) and in a pending list (DoctorPendingList) as defined in the patient struct in Figure 4.4 (c). If DID is present in DoctorGrantedList then print a message "already has access to patient data". If in the pending list then print message access request already sent else add in the pending list.

Algorithm 6 *requestDataAccess( )*

Input: DID
Output: print message or record added in pending list

1: **if** $DIDexists$ **then** ▷ checks doctor exists in BN
2: **if** $inDoctorGrantedList$ **then** ▷ checks doctor in granted list by patient
3: print already has access to patient data
4: **end if**
5: **if** $inDoctorPendingList$ **then** ▷ checks doctor in pending list
6: print access request already sent
7: **else**
8: add DID in pending list
9: **end if**
10: **end if**

iv) Grant access right by patient: Using the *grantPermission( )* function described as Algorithm 7, the patient checks DID in his pending list. If it is there, then add in DoctorGrantedList and simultaneously remove DID from the pending list. Also, update the doctor list by adding patient id PID in his patient list.

Algorithm 7 *grantPermission( )*

Input: DID
Output: grant permission to doctor

```
1: if DIDexists then
2:     patient checks for already access granted or in pending list
3:     if inPendingList then
4:         append DID to granted list and also add patient PID in doctor list
5:     end if
6: end if
```

v) Revoke function: Revoke function described in Algorithm 8, is triggered by the patient at any time. By *revoke ( )* function means the patient takes back all access rights from the entity such that he neither adds to nor views the complete medical history of patients. In this function, the patient removes the DID from the DoctorGrantedList and also updates the patient list by removing PID from the Doctor List.

Algorithm 8 *revoke( )*

Input: DID, PID
Output: revoke permissions from doctor

```
1: if DIDexists then
2:     if inDoctorGrantedList then
3:         remove DID from list
4:         also remove patient from doctor list
5:     end if
6: end if
```

Similarly, a set of functions have been implemented for TPA, such as searching already registered patient's records by PID, requesting data access, grant/revoking permissions to TPA, as defined for the patient and doctor entity earlier.

c) Blockchain Network: Blockchain Network (BN) consists of a P2P network and consensus mechanism which governs the communication over the network. In the proposed architecture, BN is private, where participants are known and are first

registered by the CA and admin. All different entities perform different functions, and the admin is responsible for configuring, maintaining, and managing BN. Admin also manages how other entities access and use the network.

d) Distributed Storage: The purpose of blockchain technology is not to manage massive amounts of transaction data. CouchDB is deployed as a state database to store large unstructured medical data while reducing redundancy throughout the entire blockchain. Along with the state database, IPFS is used to store medical reports, which improves the scalability. The EHRs are first stored on IPFS using the *ipfs add* file command, and then the corresponding hash value is stored in the blockchain. The entity that has access and has the hash value of the EHR can only get the actual file from the IPFS using *ipfs* to get the hash value.

4.4 Results and Discussion

This section presents the results of the designed framework after executing the proposed algorithms. Further, it is also shown with the help of suitable examples that how the queries (simple as well as complex) are being handled by the proposed framework.

4.4.1 Performance Evaluation and Analysis

The performance evaluation and analysis of the proposed framework is presented in this section. *Hyperledger Caliper* benchmarking tool is used for measuring the performance of the implemented blockchain network. The performance depends upon various factors such as transaction size, number of peers, block size, batch time, etc. This tool also provides different performance metrics, such as throughput and latency, to measure the performance of the system. The configuration of the system that is used for the implementation and evaluation of the proposed architecture is shown in Table 4.2.

Table 4.2: System Configuration

Component	Configuration
Processor	Intel(R) core(TM)i5 2.50GHz
Memory	8GB
Operating System	Ubuntu Linux 18.04.1LTS
Docker Engine	Version18.09.7
Docker Compose	Version1.17.1
Language	Go for Smart Contract
Hyperledger Fabric	v1.4

Different scenarios are considered for understanding, evaluating, and analyzing blockchain technology. In these scenarios, either the transaction rate varies, or the number of transactions varies. The performance is measured as transaction throughput and latency.

Transaction Throughput is the number of transactions or queries that are recorded in the ledger within a specific time frame. It is also defined as the number of total transactions committed per second (TPS) by the underlying blockchain network. In mathematical formula, it is written as:

$$T_{Th} = (T_{tc} \div T_{ts}) * n_{cn} \tag{4.1}$$

Where T_{Th} is the transaction throughput, T_{tc} is the total committed transactions (valid), T_{ts} is the total transactions time and n_{cn} is committed nodes. To get the total committed number of transactions (T_{tc}), the failed or invalid transactions are subtracted from the total transactions.

Transaction Latency is measured in seconds. It is the amount of time taken by transactions from submitted to the results available in the network. Also, it is defined as the time taken by a transaction or query from the submission by the

client until it is processed and written on the ledger. In mathematics formula:

$$T_{Lat.} = (T_c * N_{th}) - St \tag{4.2}$$

Where $T_{Lat.}$ is transaction Latency, T_c is the transaction confirmation time, N_{th} is the threshold of a network, and St is the time at which the transaction is submitted in the network.

4.4.1.1 Scenario 1: Varying Transaction Rate (tps)

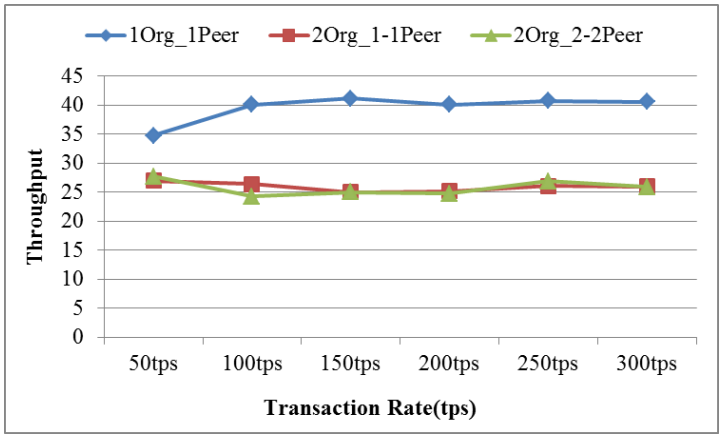
In this evaluation, various performance parameters such as throughput and latency are measured. They are measured by executing transactions on the network ledger by considering the different number of peers. The evaluation occurs in six rounds, with 1000 transactions in each round. Each round is executed at a different transaction rate of 50,100,150,200,250,300 transactions per second (tps) as defined in Table 4.3.

Table 4.3: Simulation Parameters for Scenario 1

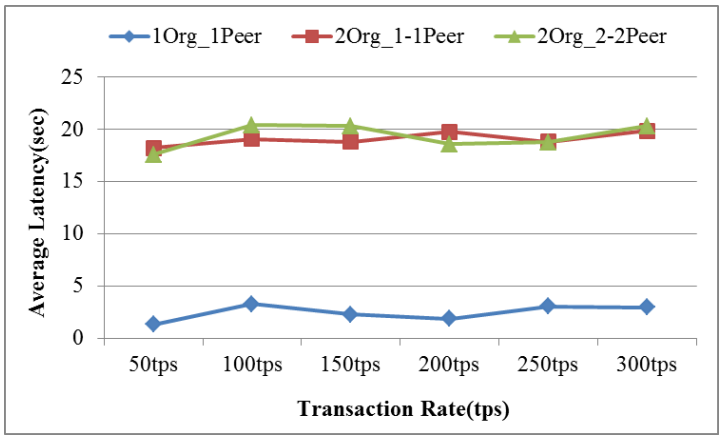
Parameters	Configuration
Rounds	6
Transactions	1000 each
Transaction Rate(tps)	(50,100,150,200,250,300)
State DB	CouchDB
Orderer	Solo(Single Host)
Size	1Org with 1Peer, 2Orgs with 1-1 peer each, 2Orgs with 2-2 peer each

Figure 4.6 shows the results in terms of throughput (TPS) and average (avg) latency in seconds (sec). Figure 4.6 (a) depicts the change in throughput, keeping the number of transactions the same, i.e. 1000 with different transaction rates for varying numbers of peers, i.e 1Org_1peer, 2Org_1-1 peer, 2Org_2-2peer each. 1Org_1peer reaches a higher throughput, i.e. 41.2 TPS, as compared to others. It is observed that with more organizations and more peers, the throughput

decreases but is improved by tuning different parameters such as block size, batch time, etc. The small decrease in throughput with more organizations and more peers shows the scalability of the system scalability.



(a) Transaction Rate vs. Throughput



(b) Transaction Rate vs. Average Latency (sec)

Figure 4.6: Impact of Variation of Transaction Rate (tps) on Throughput (TPS) and Average Latency (sec)

Figure 4.6 (b) shows the impact of the transaction rate on average latency. Low latency is a good indicator. It is observed that the latency increases with tps at the minor difference. Also, lower latency results in higher throughput. As a result, latency and throughput are inversely proportional. 1Org_1Peer has a higher throughput of 41.2 TPS and a lower latency of 1.34 sec.

4.4.1.2 Scenario 2: Variations in transaction rate (tps) and the number of transactions (tx)

The simulation parameters for the second scenario are defined in Table 4.4. In this case, the number of transactions such as 500, 1000, 1500, and 2000 is executed at the transaction rate of 50,100,150, and 200 tps each. The impact of variation of both tps and transactions on throughput and average latency is shown in Figure 4.7.

Table 4.4: Simulation Parameters for Scenario 2

Parameters	Configuration
Rounds	3
Transactions	500, 1000, 1500, 2000
Transaction Rate(tps)	50, 100, 150, 200
Size	2Org_2-2Peer each

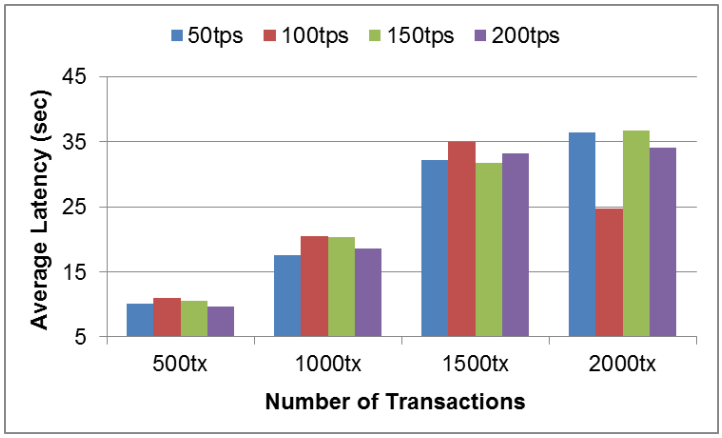
Figure 4.7 shows the impact of variation of transaction rate and number of transactions on throughput (TPS) and Average Latency (sec) in seconds. Figure 4.7 (a) depicts that throughput increases with an increase in both transaction rate (tps) and the number of transactions (tx), as observed in the graph. At 50tps, the throughput for different transactions, such as 500tx, 1000tx, 1500tx, and 2000tx, increases. Similarly, at 100tps, 150tps, and 200tps, the same pattern follows.

Figure 4.7 (b) shows that average latency decreases with an increase in the tps for the higher throughput for a particular number of transactions. As for 500tx at a different rate, throughput is higher at 200tps, and latency is lower.

4.4.1.3 Scenario 3: Performance based on function categories

The functions defined in the smart contract are categorized into two classes based on reading and writing transactions. The first one is query transactions, which combine all the functions that do read operations about retrieving data from the

(a) Throughput



(b) Average Latency (sec)

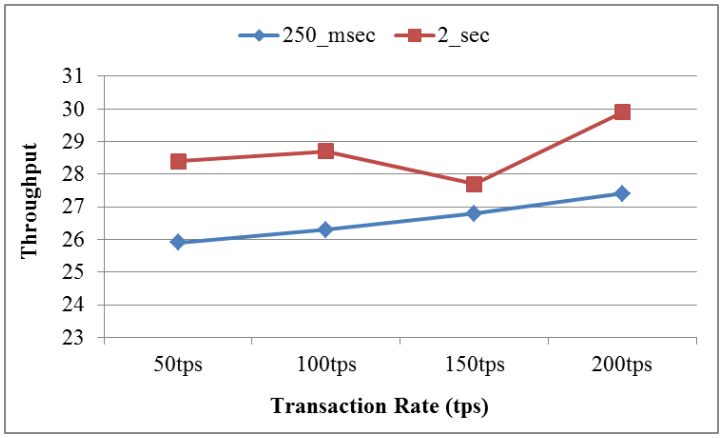
Figure 4.7: Impact of Variation of Transaction Rate and Number of Transactions on Transaction Throughput (TPS) and Average Latency (sec)

database. The second class is active transactions that combine all functions that perform write operations. The comparison in terms of throughput and average latency by tuning various parameters such as batch timeout is shown in Figure 4.8. Batch timeout here defines the time needed to wait after the first transaction arrives for more transactions before cutting a block. The evaluation configuration is shown in Table 4.5.

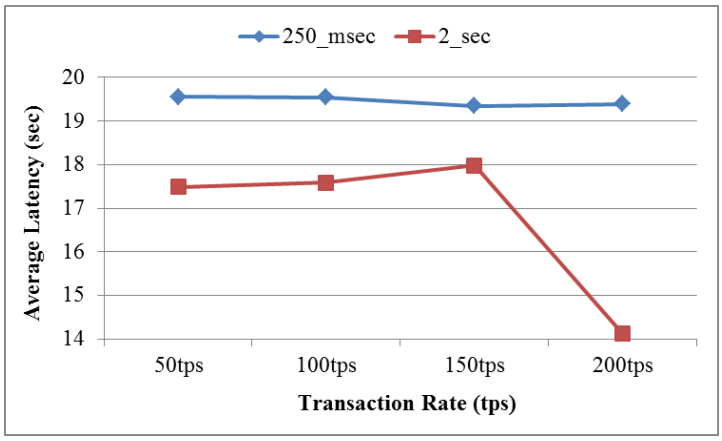
Effect of batch time on active transactions: Figure 4.8 (a) shows the effect of change in batch timeout on the performance. It is observed that the throughput (TPS) increases from 25.9 to 28.4 at 50 tps when batch time increases from the default value of 250ms to 2 sec. Similarly, in Figure 4.8 (b), the average latency decreases from 19.5 to 14.13, which enhances the system performance.

Table 4.5: Evaluation Configuration Parameters for Scenario 3

Parameters	Configuration
Rounds and Size	4 for 2Org_1-1Peer each
Transactions	1000 each
Transaction Type	Active Transactions, Query Transactions
Transaction Rate(tps)	50, 100, 150, 200
Factor Varies	Batch timeout (250_msec, 2sec)



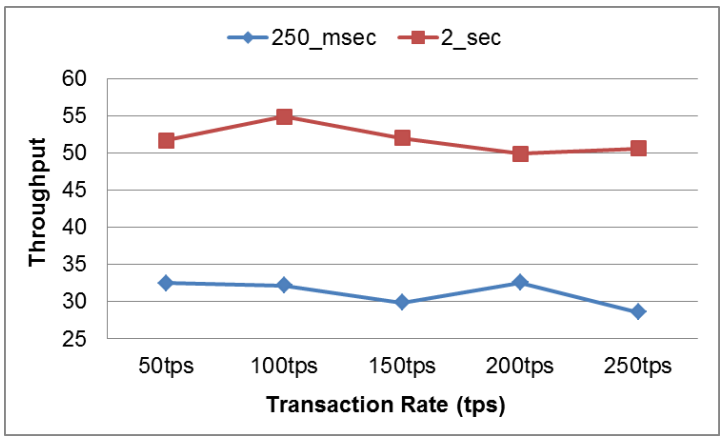
(a) Throughput (TPS)



(b) Average Latency (sec)

Figure 4.8: Impact of change in batch time on Active Transactions Throughput and Average Latency (sec)

Effect of batch time on query transactions: Figure 4.9 (a) shows the effect of change in batch timeout on the throughput and latency of query transactions. It is observed that the throughput of query transactions increases from 32.5 to 52.2 at 50 tps and 28.6 to 50.8 at 250 tps when batch time increases from default value 250ms to 2 sec.



(a) Throughput

(b) Average Latency (sec)

Figure 4.9: Impact of change in batch time on Query Transactions Throughput and Average Latency (sec)

Similarly, in Figure 4.9 (b), the average latency decreases from 6.62 to 5.46 at 50 tps and from 7.77 to 5.25 at 250tps, which enhances the system performance.

4.4.1.4 Scenario 4: Performance Comparison with other related work

The performance comparison of existing related work[86,88] and proposed work is demonstrated in the form of graphs and tables as shown in Figure 4.10, Figure 4.11, Figure 4.12, Figure 4.13, Table 4.6, and Table 4.7. For this comparison, only those related research papers are considered that use the same implementation platform and have a similar set of assumptions and constraints for securing and sharing healthcare data. Table 4.6 highlights the performance comparison of proposed and existing work based on the number of transactions completed w.r.t. time (sec).

Table 4.6: Performance Comparison of proposed and existing work based on no. of the transaction completed w.r.t. time (sec)

Criteria → Ref. ↓	**No. of active/write Transactions completed w.r.t. time(sec)**	**No. of all Transactions completed w.r.t. time (sec)**
Tanwar et al. [86]	500, 1000, 1000, 2000, 2000, 3000 (approx.) at time (in secs) 40,60,80,100,120,140 respectively.	Not Performed
Koushik et al. [88]	Not Performed	Not Performed
Proposed Work	618, 968, 1354, 1825, 2253, 2810 (approx.) at time (in secs) 40, 60, 80, 100, 120, 140 respectively.	669, 979, 1238, 1783, 2049 at the time (in secs) 40, 60, 80, 100, and 120, respectively.

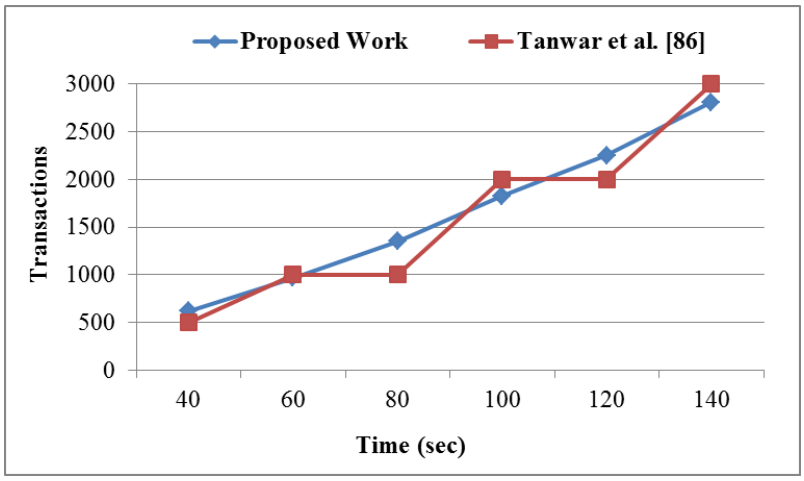


Figure 4.10: Number of active transactions completed w.r.t. time

Figure 4.10 shows the comparison between the proposed work and related work [86] based on the number of transactions completed with time. It is observed that the performance of the proposed work is better or comparable in some cases with the related work. Figure 4.10 depicts that the system [86] takes 120 seconds to complete 2000 transactions (approx.), whereas the proposed model completes 2253 transactions. Similarly, in 140 seconds, 3000 transactions are completed in the related work, while the proposed system completes 2810 transactions.

Table 4.7 highlights the performance comparison of proposed and existing work in terms of throughput and latency on the basis of three performance parameters: (i) *types of transactions (active and query)*, (ii) *number of transactions* and (iii) *the transaction rate*. Further, the performance is compared on the basis of the types

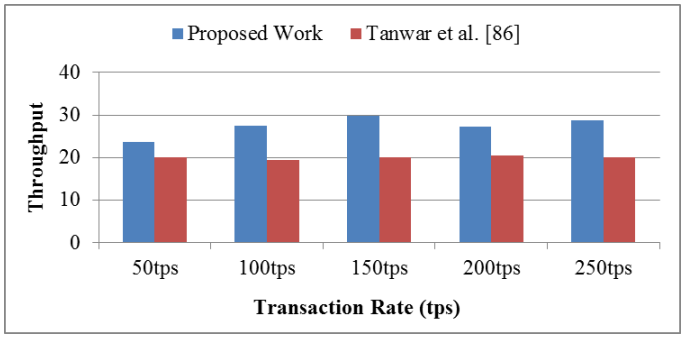
Table 4.7: Performance Comparison of proposed and existing work based on query handling in terms of throughput and latency

Criteria → Ref. ↓	**Type(s) of Transaction**	**Number of Transactions (tx)**	**Transaction Rate (tps)**	**Throughput (TPS) [Min-Max]**	**Average Latency (sec) [Min-Max]**
Tanwar et al. [86]	Write	1000	50-250	19-21	29-31
	Read	1000	100-200	50-52	6-9
Koushik et al. [88]	Write and Read	10, 100	2.8-4.2	1.3-9.21	
Proposed Work	Active/Write	1000	50-250	**Simple:** 50-67.7 **Complex:** 23.7-29.9	**Simple:** 0.27-3.02 **Complex:** 14.13-17.98
	Query/Read	1000	100-200	**Simple:** 49.9-68 **Complex:** 49.9-54.9	**Simple:** 0.21-0.48 **Complex:** 4.29-5.63

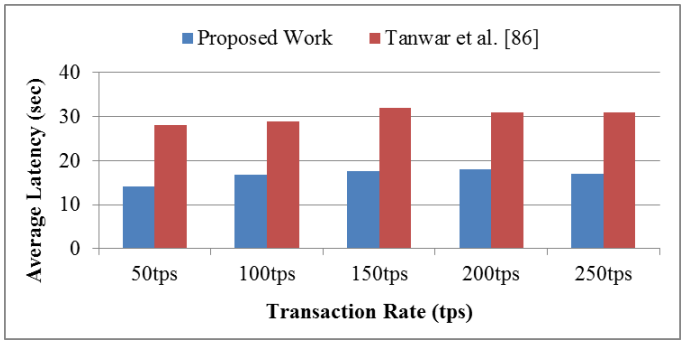
of queries handled by the system. In the proposed framework, the throughput and latency range for both types of queries (simple and complex) as handled by the system is shown. The comparison is made on the assumption that the existing work [86] also supports complex query handling, as the authors have not mentioned the type of queries they executed.

Figure 4.11 and Figure 4.12 demonstrate the performance Comparison of proposed and existing work [86] based on throughput and average latencies of active and query transactions in terms of query handling. Figure 4.11 (a) and Figure 4.11 (b) depict that the proposed system performs better in terms of throughput and average latency of active/write transactions in comparison to the existing approach [86].

Figure 4.12 (a) and Figure 4.12 (b) show that the proposed framework performs better considering throughput and average latency of query/read transactions in comparison to the existing approach [86].

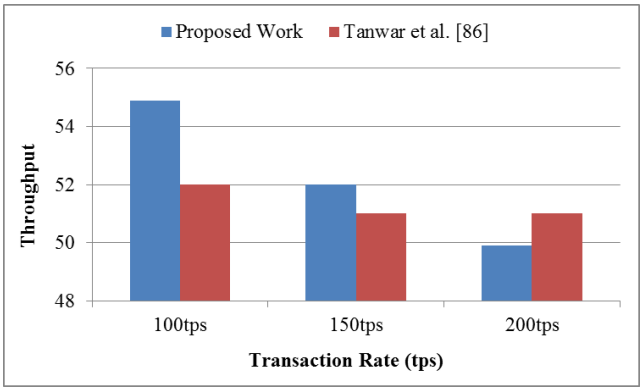


(a) Throughput of active/write transactions

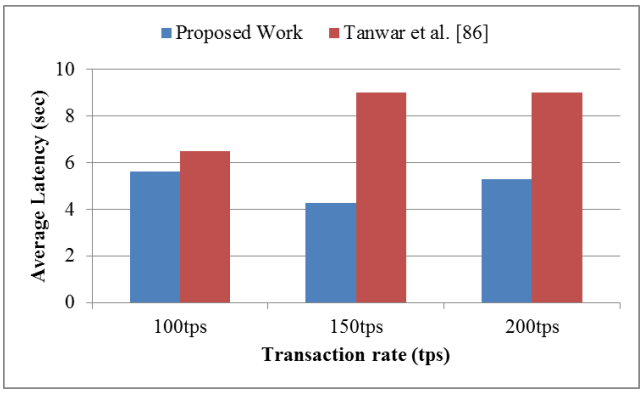


(b) Average Latency (sec) of active/write transactions

Figure 4.11: Comparison of proposed and existing work[86] based on active transactions in terms of throughput and average latency

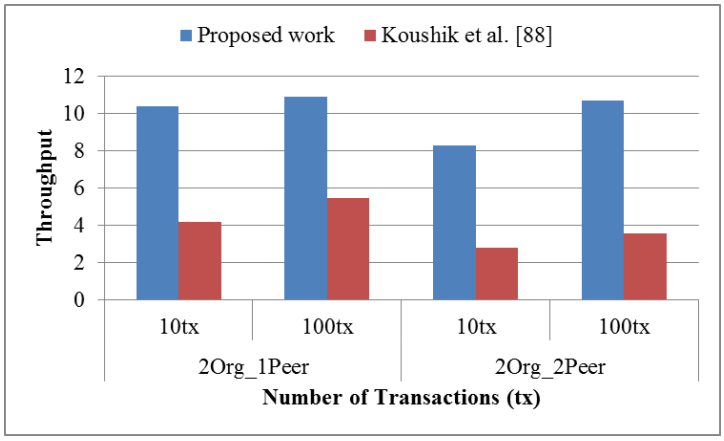


(a) Throughput of query/read transactions

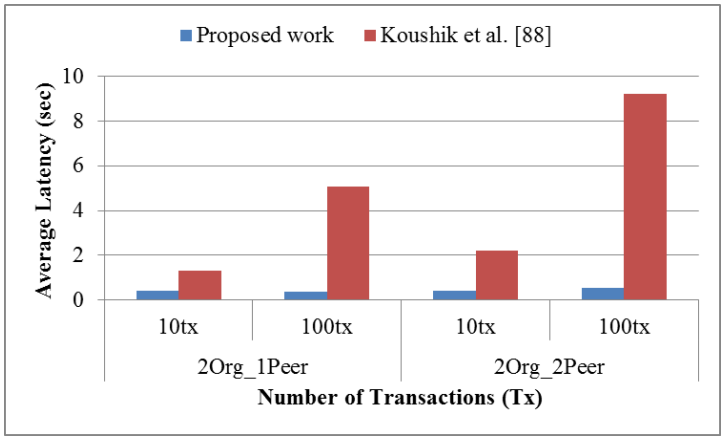


(b) Average Latency (sec) of query/read transactions

Figure 4.12: Comparison of proposed and existing work[86] based on query transactions in terms of throughput and average latency



(a) Throughput



(b) Average Latency (sec)

Figure 4.13: Throughput and Average Latency (sec) comparison between proposed and existing work[88]

Figure 4.13 shows the comparison in terms of throughput and average latency (sec) considering 2Org_1Peer and 2Org_2Peer. Figure 4.13 (a) and Figure 4.13 (b) depict that the proposed system performs better, more than twice the existing work[88] with very little change in latency. The existing work[88] has Min Avg. Latency: 1.3sec at 10tx with 2Org_1Peer and Max Throughput: 4.2TPS at 10tx with 2Org_1Peer while the proposed work achieves Min Avg. Latency: 0.36sec at 100tx with 2Org_1Peer and Max Throughput: 10.9TPS at 10tx with 2Org_1Peer.

4.4.2 Assessment of the Proposed Framework using Queries

1. *Query-based on Key only:* Figure 4.14 shows the example of Query-based on Key only by Retrieving all records of a doctor with dID is “D001”.

2. *Query-based on Key and data values:* Figure 4.15 shows the example of Query-based on Key and data values in which we retrieve the ID, name, and gender of patients with blood group: “AB-” from all documents.

3. *Query-based on the master key and derived composite keys:* Display all documents where the disease value is Thyroid, demonstrates the example for Query-based on the master key and derived composite keys shown in Figure 4.16.

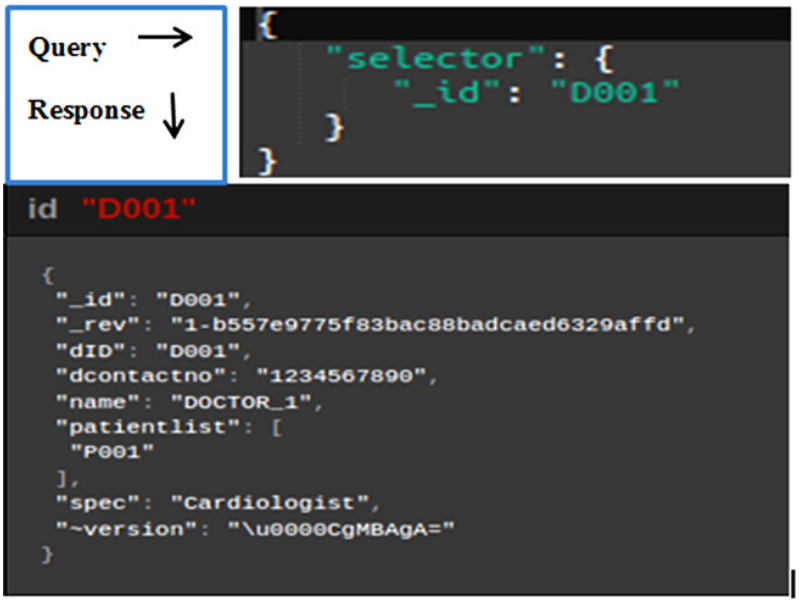


Figure 4.14: Example of Query based on Key only

Figure 4.15: Example of Query based on Key and data values

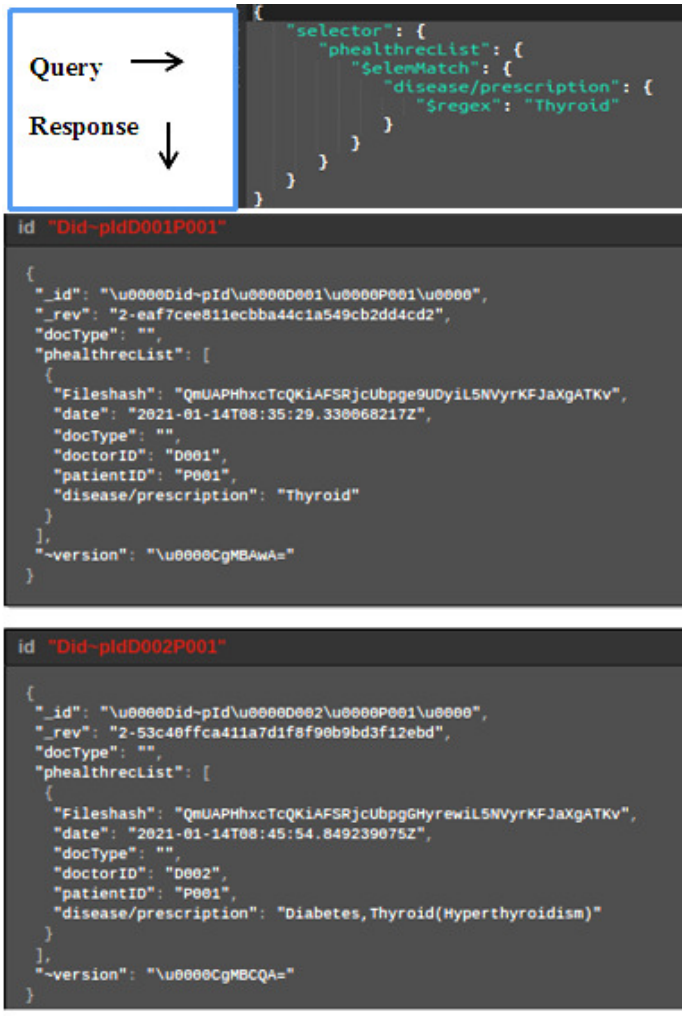


Figure 4.16: Query based on the master key and derived composite keys

4.5 Chapter Summary

This chapter explores the necessary tools, platforms, and databases needed to establish the groundwork for creating and implementing the proposed framework. In this chapter, a permissioned blockchain-based framework involving multiple healthcare entities to securely store, share, and query EHR data is designed and implemented. The algorithms are defined in terms of smart contracts to describe the functionality of patients, doctors, and TPA's. For storing unstructured medical data and for handling rich queries, a NoSQL document-oriented database *CouchDB* is used. Security of the system is achieved through the tamper-proof, immutable ledger, as unauthorized users are unable to modify the ledger. Finally, the performance of the system is evaluated using a *Hyperledger Caliper* tool for different configurations based on transaction rates and transaction size. The experimental results indicate that while increasing the number of peers during simulation, there is a slight decrease in throughput. Thus the proposed framework is capable of being scaled, and its performance can be enhanced through tuning of parameters. The comparison between different types of transactions is done by varying batch times. There is a significant increase in throughput and a decrease in latency for both active and query transactions by varying the batch time. Also, the performance comparison of existing related work[86,88] and proposed work is demonstrated. Further, the system is also evaluated for the handling of complex queries with examples. The following chapter explores the enhancement of security and privacy through integration of cryptographic algorithm IB-PRE in the proposed framework.

Chapter 5

AN APPROACH FOR ENHANCING SECURITY AND PRIVACY OF HEALTHCARE DATA

In this chapter, the Identity-Based Proxy Re-Encryption (IB-PRE) algorithm is utilized to enhance the privacy and security of healthcare data. The integration of the IB-PRE algorithm into a permissioned blockchain-based framework for healthcare data ensures the secure sharing of records while preserving privacy by avoiding the exposure of private keys. Several experiments on real-time medical datasets to validate and assess the performance of the proposed framework using distributed storage are conducted. Features and performance-based comparisons of the proposed and existing related work(s) are made to demonstrate the effectiveness.

5.1 Motivation

The literature review highlights the critical demand for enhanced security and privacy in healthcare data sharing, through blockchain technology. It highlights the limitations of existing data-sharing schemes, especially in managing and access controlling of EHRs. This gap underlines the necessity to explore and apply advanced cryptographic techniques, such as Proxy Re-Encryption (PRE), within blockchain frameworks in the healthcare domain. The authors[114] discuss three

data storage scenarios: plain text, ciphertext, and hash value. To enhance privacy, they propose cryptographic techniques like symmetric encryption, attribute-based signcryption, homomorphic encryption, and anonymous authentication. Using digital signatures and Authenticated Key Exchange (AKE), the authors suggest a model to preserve privacy for accessing real-time medical services[115]. PRE algorithms are recognized for their efficacy in secure record sharing on external platforms like cloud servers[116], making them an ideal candidate for privacy preservation in healthcare data while sharing. The authors of[117] proposed a framework based on PRE to control access while improving data sharing efficiency and preserving privacy.

However, the system suffers from some limitations, such as limited storage capacity, use of cryptographic algorithms, and handling of simple queries, among others. This highlights the necessity for the development of innovative solutions to enhance the security and privacy of healthcare data management within blockchain frameworks.

To address the above issues, a permissioned blockchain with IB-PRE[118] is integrated with the proposed framework. This integration aims to achieve the following objectives.

- For privacy preservation and efficient management of EHRs, a blockchain-based framework with IB-PRE is proposed, which includes all health sector players, i.e., Regulatory Authority (RA), doctors, patients, and other TPAs.

- An algorithm is proposed and designed using smart contracts for providing access permissions to health sector players to create, update, and share secured healthcare data.

- To enhance system scalability and reduce communication time, a hybrid approach for EHR storage is proposed. The encrypted records are stored

off-chain (IPFS), while hashes are stored on-chain (blockchain).

Table 5.1 shows a feature-based comparison of proposed work with existing systems[119,120,121]. The features included are the language used for designing smart contracts, a storage repository for storing massive medical records, and a proxy re-encryption algorithm based on identity for securely sharing encrypted records placed at distributed storage without exposing the plain text, complex query handling, and usage of NoSQL Database for handling unstructured data. Go language over Python for smart contracts is chosen because it is quicker, has built-in support for concurrency, and can handle extensive, scalable programs. To make the system more scalable, off-chain distributed storage as a repository over Blockchain for storing massive EHRs is preferred.

Table 5.1: Feature-based Comparison of Proposed and Existing Blockchain-based EHR System

Features → Reference(s) ↓	**Designing of Smart Contract**	**Distributed Storage as repository**	**Usage of PRE algorithm**	**Complex Query Handling**	**Performance Analysis**	**NoSQL Database**
[119]	Go	IPFS	×	√	√	CouchDB
[120]	Python	Blockchain	×	-	√	×
[121]	Python	Blockchain	×	-	√	×
Proposed Work	Go	IPFS	√	√	√	CouchDB

5.1.1 Importance of Integration of IB-PRE Algorithm with Proposed Framework

Consider the case where there is no Proxy Node, i.e., if several requesters have made the same record request, the patient must re-encrypt the records each time using the requester's public key and upload them to IPFS for distribution. This process may result in storage waste and increased communication costs. Here, a proxy node is used to solve these issues, which decreases overhead on the patient side and the storage waste in the IPFS network. Imposing encryption on records is

also a computationally intensive task. Using a smart contract to implement these encrypted records causes high computational costs. Thus, the proxy re-encryption node is utilised to access data from IPFS and perform aggregate operations acting as an intermediator for secure data sharing.

5.2 Preliminaries and Background

5.2.1 Bilinear Pairing

Let G_1, G_T be the Groups of the same prime order p and map $e : G_1 \times G_1 \longrightarrow G_T$ is bilinear mapping if satisfies following features:

i) Bilinearity: Given $g \epsilon G_1$, and $\forall \alpha, \beta \epsilon Z_p^*$, the equation holds $e(g^\alpha, g^\beta) = e(g,g)^{\alpha\beta}$.

ii) Non Degeneracy. (i.e., if $G_1 = \langle g \rangle$, then $G_T = \langle e(g,g) \rangle$)

iii) Mapping e is computed efficiently.

5.2.2 Decisional Bilinear Diffie Hellman Problem (DBDH) Hardness Assumption

Let (G_1, G_T) be a pair of bilinear groups having with mapping $e : G_1 \times G_1 \longrightarrow G_T$ which is efficiently computable, and g be a random generator of G_1. The DBDH problem is to decide, given a tuple of values $(g, g^\alpha, g^\beta, g^\gamma, T) \epsilon G^4 \times G_T$ (where $\alpha, \beta \epsilon_R Z_p^*$), whether $T = e(g,g)^{\alpha\beta\gamma}$ or if T is a random element of G_T [118].
Formally, the DBDH assumption holds good if $\forall$ probabilistic polynomial-time algorithms $\in$ is negligible.

$$\Pr[\alpha, \beta, \gamma \leftarrow Z_p^*; 1 \leftarrow \hat{A}(g, g^\alpha, g^\beta, g^\gamma, e(g,g)^{\alpha\beta\gamma})]\text{-}$$
$$\Pr[\alpha, \beta, \gamma \leftarrow Z_p^*; T \leftarrow G_T; 1 \leftarrow \hat{A}(g, g^\alpha, g^\beta, g^\gamma, T)] \leqslant \in$$

5.2.3 Identity Based Proxy Re-Encryption (IB-PRE)

Green and Ateniese[118] have proposed IB-PRE, which permits a partially trusted proxy to reconstruct the ciphertext encrypted with user A identity (public ID) to target ciphertext for user B without exposing plaintext. Then user B uses his private key for final decryption to get the actual text. During the entire communication, the proxy cannot acquire any information related to plaintext. A PRE comprises of six polynomial-time functions, namely Setup(), Key_Gen(), Encrypt(), Decrypt(), RKGen(), and ReEncrypt().

5.3 Proposed Framework and Methodology

This section presents the proposed framework's architecture, working, and implementation details. The proposed system comprised of various entities such as patients, doctors, proxy-node, Key Generation Center (KGC), and requesters. All the entities must register with the Regulatory Authority to join the blockchain network. Smart Contracts/ Chaincodes[122] are designed and implemented using Golang[113] for executing coordinated transactions and distributed file systems, i.e., IPFS as off-chain storage is used for securely storing EHRs. Additionally, all the data is saved in a distributed ledger, which is subsequently utilised for authentication. The various symbols/abbreviations used are listed in the Table 5.2 with their description.

5.3.1 Architectural Design and Functionality of Proposed Framework

Figure 5.1 presents the architectural design of the proposed framework after integration with IB-PRE algorithm, which consists of several components. The functionality of the architecture is divided into three main phases.

Phase 1: Registration and Authorization During Phase 1, all entities request

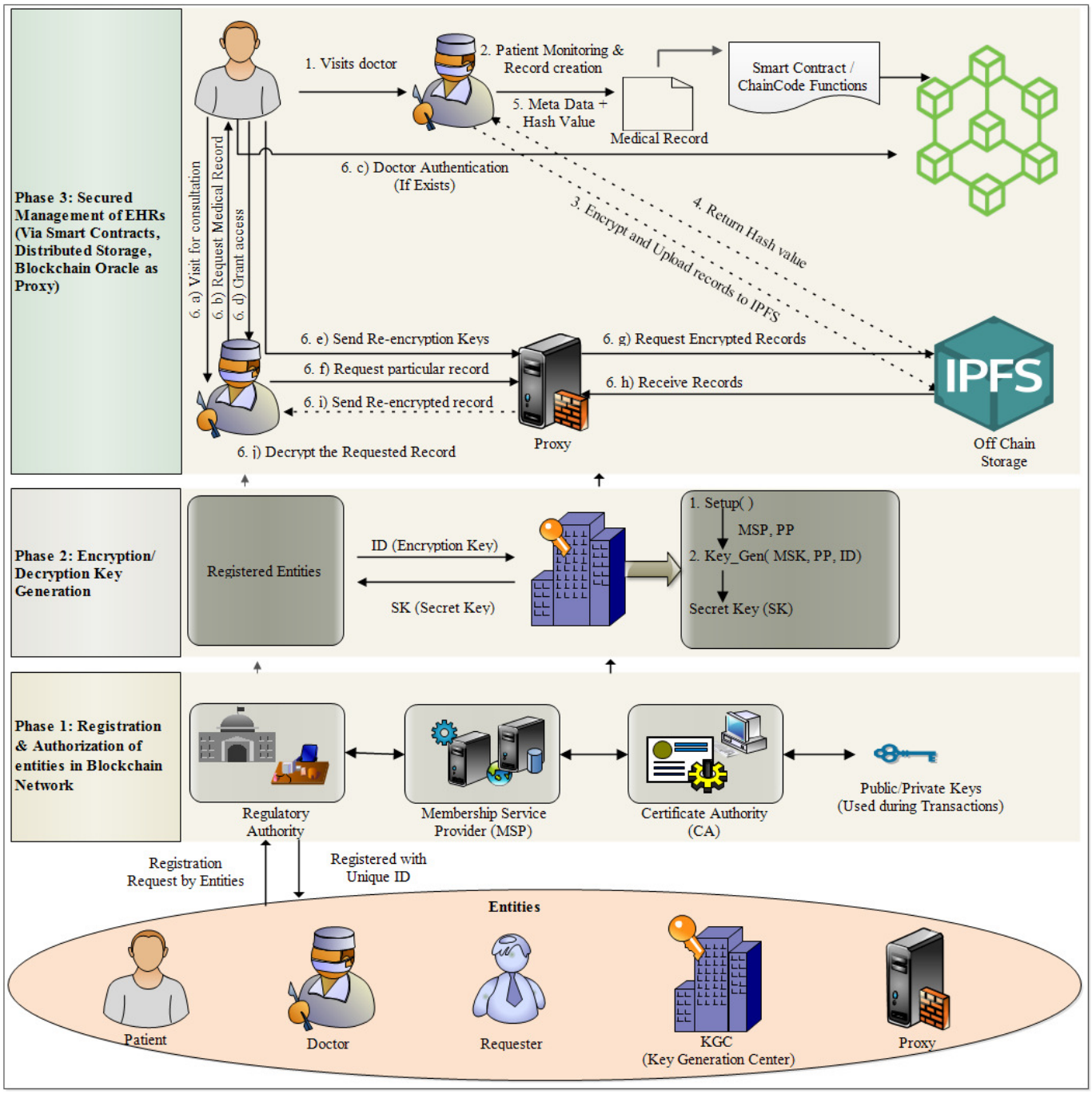


Figure 5.1: Architecture of the Proposed Framework

Table 5.2: List of Symbols

Symbol	Description
P_{id}	Denotes Patient id
$PEmail$	Email id of the Patient
D_{id}	Denotes Doctor id
R_{id}	Denotes Requester id
MSK	Master Secret Key kept secret by the admin
pp	Public Parameters
id_i	Public ID of user $i, i = 1..n$
sk_{id_i}	Secret ID of user corresponding to Public ID id
$mrecord$	Medical Record of Patient
$C1_{mrecord}$	first level cipher text
$C2(C1_{mrecord})$	second level cipher text
$rk_{id1 \rightarrow id_2}$	re-encryption key from id1 to id2

Regulatory Authority (RA) for registration. RA responds to the requested user with the Certificate Authority (CA) certificate via the Membership Service Provider (MSP). It includes a unique Public Identity (ID) key and a private key. Then, all the information is updated in the blockchain network.

Phase 2: Encryption/Decryption Key Generation In Phase 2, all registered entities use KGC to generate identity-based encryption-decryption keys. The entities send their public identity (email-id, ID) to KGC and receive a secret key corresponding to the identity. Entities will later use this pair of encryption-decryption keys during secure EHR sharing.

Phase 3: Secured Management of EHRs During Phase 3, secured EHR management while preserving user privacy will be done using designed smart contracts, distributed storage, and Proxy Node. The schema of the EHR includes the following fields, which are defined in the smart contract:

- PatientID: It is a unique ID assigned to each patient during the registration process.
- Doctor: It is a unique ID of a doctor who creates the patient EHR.

- Prescription: It stores prescribed medication. Multiple values separated by commas can also be inserted.

- FileHashes: It holds the hash value of a record's data, such as lab tests, X-rays, etc., placed in the distributed storage.

- Date: It is the timestamp indicating when the medical record is generated and preserved within a blockchain system.

For example, Consider the case of record creation. The doctor monitors the visited patient and creates EHRs with permission by invoking chain code functions. The doctor encrypts the records with patient ID (public) using the *Encrypt()* function and uploads the corresponding ciphertext to IPFS. The hash value is returned as the response to the uploaded file and then added to the ledger.

Now, consider a case where the doctor (requester) requests historical records from the patient during the consultation. After authentication from the blockchain, the patient grants access to the requester and informs the proxy node. Also, create and send re-encryption keys to the proxy node by executing the *RKGen()* function. Proxy Node has access to the ledger so it can check the authenticity of the patient and requester. After authentication, gets actual records as ciphertext from IPFS. After applying the *Re-encrypt()* function on the received ciphertext, it generates the second-level ciphertext (final) for the requester. Finally, the final ciphertext is transmitted to the requester, which he decrypts using a private key by executing *Decrypt()*function.

5.3.2 Components of the Proposed Framework

The main components, such as entities (patients, doctors, requesters, KGC, Proxy Node), Smart Contracts, and IPFS that constitute the proposed model shown in Figure 5.1 are discussed below, and the interaction among these components is illustrated in Figure 5.2.

i) ***Regulatory Authority:*** A government or trusted official who is responsible for registration and authorization of different entities such as doctors, patients, proxy node, and TPA participating in the network. As per the Indian context, for the proposed healthcare system, the RA can be the Medical Council of India (MCI) or the National Medical Commission (NMC), which are responsible for overall licensing and regulating medical education and practice in India.

ii) ***Patients:*** A patient is an entity in the blockchain network that visits various doctors for treatment or consultation. The patient is also the content owner, as the record created belongs to him/her. A patient grants/ revokes access to doctors for creating and uploading encrypted EHRs to IPFS. The grant/revoke access to EHR is totally in the control of that patient. Patients grant access to doctors during consultation or treatment. In our implementation we have considered a window of 24 hours, i.e. data is made available to the concerned doctor for a period of 24 hours maximum only, except in the case if a patient does not revoke back early. After 24 hours of using the concept of ageing, the link to that medical record expires automatically. The patient is free to revoke or extend the access duration according to his/her treatment. Algorithm 9 describes the functionalities of the patient entity.

iii) ***Doctors:*** An entity that is responsible for monitoring, creating, and uploading the patient's EHRs to IPFS. The records are encrypted by doctors using the patient's public key.

iv) ***Requesters:*** Any registered doctor, physician, lab technician, and TPA requested for encrypted records are Requesters. They get Re-encrypted EHRs through the proxy node and then re-decrypt them locally.

v) ***Proxy Node:*** The entity that is responsible for sharing EHRs between

Algorithm 9 *Functionalities of Patient Entity*

Input: $P_{id}, D_{id}, PEmail$

procedure PATIENT(P_{id})
  if $P_{id} exists$ **then** ▷ checks if Patient with $P_{id} \epsilon BN$
    Can ViewRecords, queryRecordByDate, queryRecordsByDID
  else
    Display "P_{id} not exists"
  end if
  if $VisitDoctor$ **then**
    if $D_{id} exists$ **then** ▷ checks if Doctor with $D_{id} \epsilon BN$
      checkPermission
      if $granted$ **then**
        Send Public ID for encrypting EHRs
        Can Revoke Permission any time
      else
        grantPermission to Doctor with D_{id}
      end if
    else
      Display "D_{id} not exists"
    end if
  end if
  if Any Request from Requester with R_{id} **then**
    if $R_{id} exists$ **then** ▷ checks R_{id} exists in BN
      Generate and Send Re-encryption Keys to proxy node
    end if
  end if
end procedure

delegator (patient) and delegate (doctor) using identity-based proxy re-encryption. Public IDs are used by the delegator for the generation of re-encryption keys, which are then provided to proxy by patients for re-encrypting the first-level cipher text requested by the doctor. Then, the proxy gets the requested records from IPFS, re-encrypts them using re-encryption keys, and finally transmits the Re-encrypted records to the Requester (doctor).

vi) ***Distributed off-chain storage:*** Encrypted EHRs are stored on off-chain storage, which is IPFS. The encryption is done before uploading EHRs to IPFS to prevent unauthorized access.

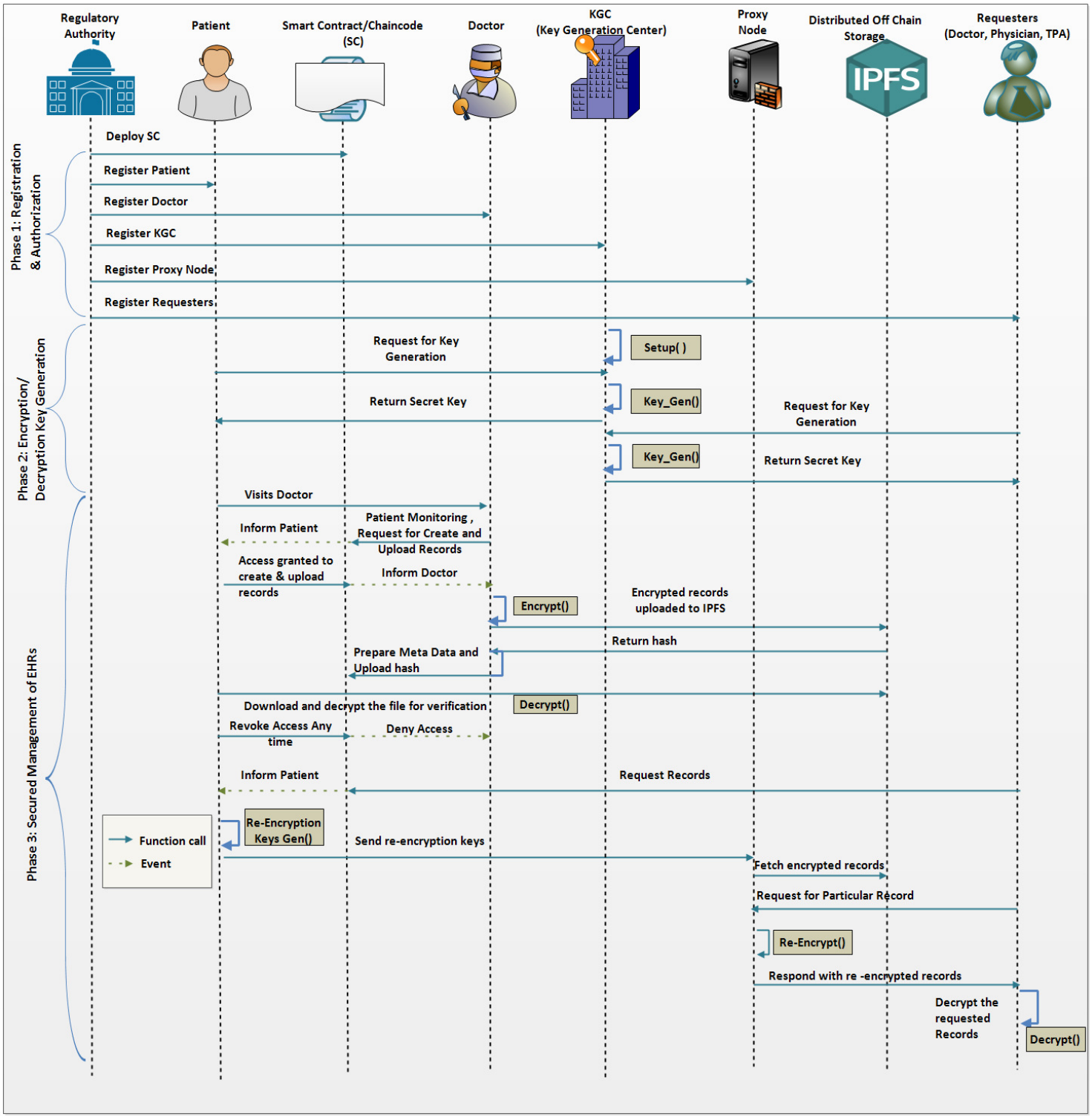


Figure 5.2: Interaction Among Different Components of the Proposed Framework

5.4 Proxy Re-Encryption Algorithm

The process of converting ciphertexts of the delegator into that of the delegatee without exposing plaintext to proxy is known as Proxy Re-Encryption (PRE). In our work, the Identity Based PRE algorithm (IB-PRE) introduced by[118] is employed that works on the assumption that DBDH is hard to decide. The IB-PRE is implemented using the charm-crypto tool[123]. The IB-PRE algorithm contains the following six functions:

1. $Setup() \Rightarrow (MSK, pp)$: The Setup() function is the system initialization step executed by the admin node. As a result the output after execution is Master Secret Key (MSK) and Public Parameters (pp).

2. $Key\_Gen(pp, MSK, id) \Rightarrow sk_{id}$: The secret/ private $Key\_Gen(pp, MSK, id)$ function takes public parameters pp, master secret key MSK, and public key id of a user as input, and then generates a secret key sk_{id} as an output.

3. $Encrypt(pp, id, msg) \Rightarrow C_1(msg)$: The $Encrypt(pp, id, msg)$ function takes public parameters pp,the identity (public) id of a user and the message msg to be encrypted as input and generates first level ciphertext $C_1(msg)$as output.

4. $RKGen(pp, sk_{id1}, id_1, id_2) \Rightarrow rk_{id1 \rightarrow id2}$: $RKGen(pp, sk_{id1}, id_1, id_2)$ function takes pp, sk_{id1} of user 1 and public key of user 1 id_1 and user 2 id_2 as input, then outputs the re-encryption key $rk_{id1 \rightarrow id2}$. The re-encryption key is used for re-encrypting a cipher text generated using an id_1 such that it is decrypted by using the secret/ private key that corresponds to an id_2.

5. $Re - Encrypt(pp, rk_{id1 \rightarrow id2}, cid1) \Rightarrow cid2$: The $Re - Encrypt(pp, rk_{id1 \rightarrow id2}, cid1)$ function is executed by the proxy by taking public parameters pp, re-encryption key $rk_{id1 \rightarrow id2}$, first cipher text $cid1$ as input parameters and generated second level cipher text $cid2$ as output.

6. $Decrypt(pp, sk_{id}, cid)$: Finally, the $Decrypt(pp, sk_{id}, cid)$ function that is executed by the delegatee by providing public parameters pp, secret key sk_{id} and cipher text cid as input and generate the plain text.

Working of IB-PRE after deploying Smart Contract and registering users (patients, doctors, proxy, TPA's) is shown in the form of pseudocode as Algorithm 10.

Algorithm 10 *Procedure Flow of IB-PRE*

Step 1⇒ {*Initialization:* System Setup() ▷ execute by admin
Output:MSK, pp}
Step 2⇒ **Key_Gen()** { *Input:* pp, MSK, id_1
Output:sk_{id_1}, pp}
Step 3⇒ **Encrypt()** { *Input:* $pp, id_1, mrecord$
Output:$C1_{mrecord}$}
Step 4⇒ **Upload()** { *Input:* $C1_{mrecord}$
Output:$Hash(C1_{mrecord})$}
Step 5⇒ **RKGen()** { *Input:* $pp, sk_{id_1}, id_1, id_2$
Output:$rk_{id1 \to id_2}$}
Step 6⇒ **Re-Encrypt()** { *Input:* pp, $rk_{id1 \to id_2}$, $C1_{mrecord}$
Output:$C2(C1_{mrecord})$}
Step 7⇒ **Decrypt())** { *Input:* pp, sk_{id_2} , $C2(C1_{mrecord})$
Output:$ActualRecords$

5.5 Performance Evaluation

In this section, the performance of the proposed architecture by running numerous experiments is conducted. *Hyperledger Caliper*, a performance benchmarking tool for experimental evaluation and to assess the performance of the proposed framework, is employed. The major parameters for determining performance are throughput in TPS and latency in seconds.

Table 5.3 shows the system configuration and simulation parameters used for the overall implementation and evaluation. In addition, the variety of factors influence system performance, including transaction rate, block size, number of transactions, etc, also analyzed.

In experimental evaluation, two organizations with one peer each are considered. The peer is a node in the blockchain network that hosts and maintains the ledger.

Table 5.3: System Configuration and Simulation Parameters

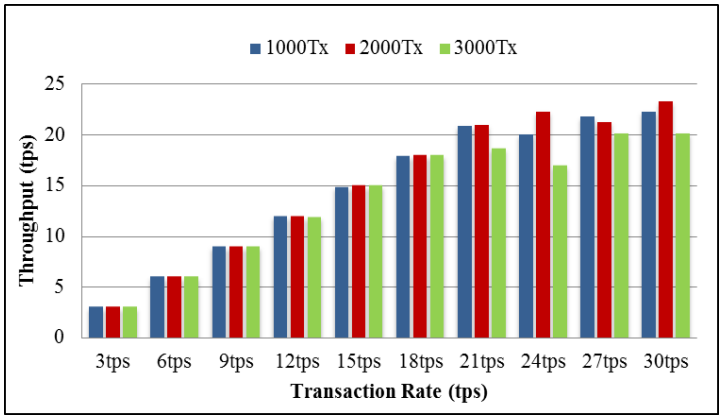
Parameter	Configuration
Processor	Intel Core-i5
Memory	8GB
Operating System	Ubuntu 18.04.1
Hyperledger Fabric	v1.4
Rounds	10
Transactions	1000, 2000 and 3000
Transaction Send Rate(tps)	(3,6,9,12,15,18,21,24,27)
State DB	CouchDB
Orderer and Size	Solo and 2Org_1Peer each

Further, peer contains a number of clients that act as end-users or entities. Clients can be patients, doctors, TPA's or requesters depending upon the smart contract we have executed. These clients communicate with the blockchain via peers. Clients can vary in number, ranging from 5 to 50.

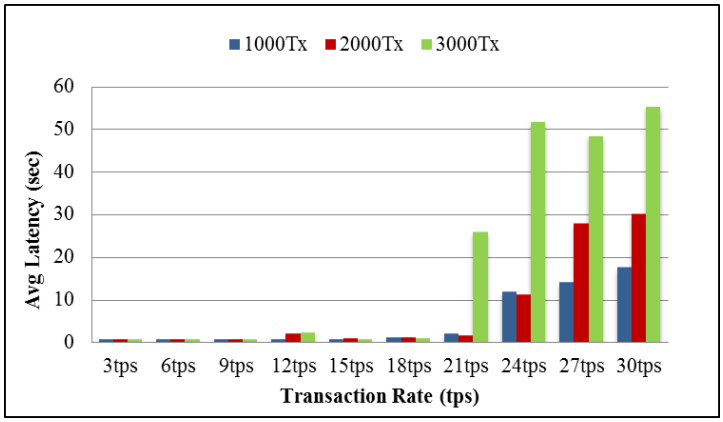
Scenario 1: Impact of varying transactions size(Tx) and rate (tps) on throughput and latency. Figure 5.3 shows the effect on system throughput and latency with the change in transaction rate (tps) and the number of transactions (Tx).

From Figure 5.3(a), it is observed that system throughput rises at a similar rate up to the transaction rate of 18 tps. Whereas, with an increase in tps, the throughput decreases slightly. Similarly, Figure 5.3(b) depicts that latency (average) rises with an increase in number of transactions (Tx) above 21tps. The system's throughput and latency can be further improved by tuning different parameters or designing optimized smart contracts.

Scenario 2: Transactions Completed w.r.t. Transaction Duration (TxDuration) in seconds. This scenario discusses the number of transactions completed in a particular duration of time (TxDuration). The configuration parameters considered for simulation are presented in Table 5.4, and the results are presented graphically in Figure 5.4. Figure 5.4(a) and Figure 5.4(b) show that the number



(a) Transaction Rate vs Throughput



(b) Transaction Rate vs Avg Latency

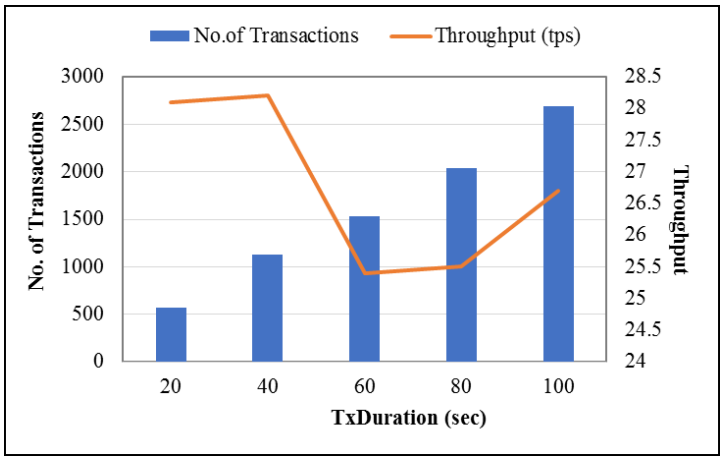
Figure 5.3: Impact of varying transactions size(Tx) and rate (tps) on throughput (TPS) and latency (sec)

Table 5.4: System Configuration and Simulation Parameters for Scenario 2

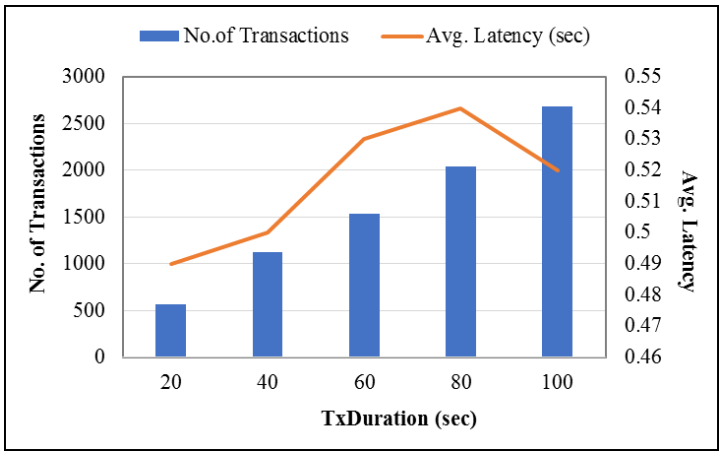
Parameter	Configuration
Rounds	5
TxDuration (sec)	20,40,60,80,and 100
Transaction Send Rate (tps)	22tps to 30tps
Orderer and Size	Solo (Single Host) and 2Org with 1Peer each

of transactions (tx) completed in 100 TxDuration (sec) are highest, i.e., 2684 transactions (tx). The max and min throughput are 28.2 (tps) and 25.4 (tps), having max and min latency of .54 sec and .49 sec, respectively.

Scenario 3: Impact of Batch Size on Throughput and Average Latency. Batch Size and Batch Timeout are critical elements that influence system throughput and



(a) No. of Transactions completed and corresponding Throughput w.r.t. TxDuration



(b) No. of Transactions completed and corresponding Avg. Latency w.r.t. TxDuration

Figure 5.4: No. of Transactions completed and corresponding Throughput and Avg. Latency w.r.t. TxDuration

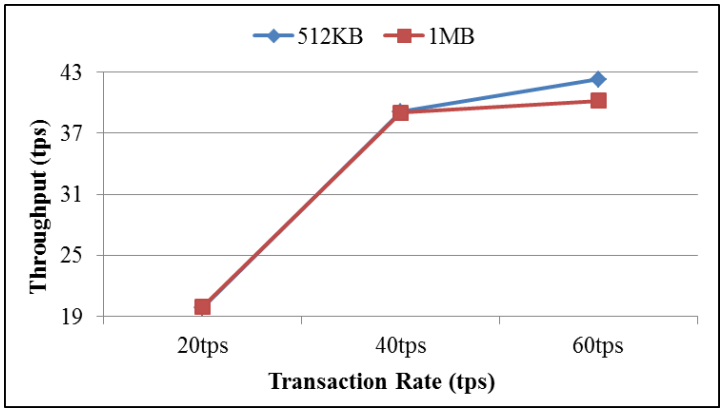
latency. Batch Timeout is specified as the duration of time to wait before creating a batch. The batch size determines how many transactions are grouped together in a block. It depends upon three factors: Max Message Count, Absolute Max Bytes, and Preferred Max Bytes. The considered simulation parameters are given in Table 5.5. Figure 5.5 depicts an analysis of the effect of Batch Size on various Preferred Max Bytes.

Figure 5.5(a) and Figure 5.5(b) demonstrate the impact of variation in Batch Size (Preferred Max Bytes) on throughput and latency. It is observed that there is a minor change in the throughput. The throughput at 512KB is more than 1MB,

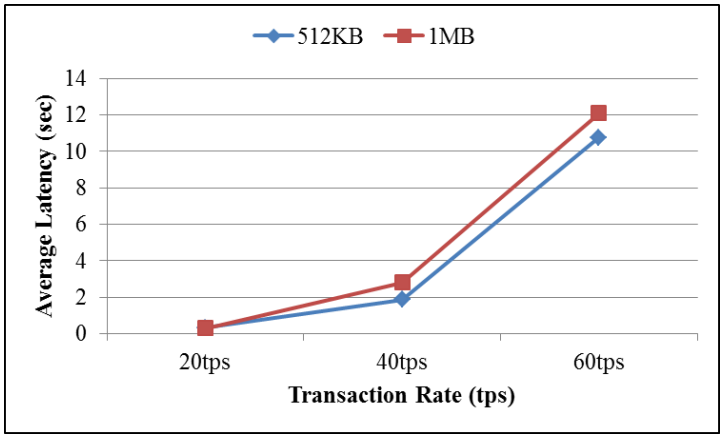
whereas the corresponding average latency in terms of seconds decreases.

Table 5.5: Simulation Parameters for Scenario 3

Parameter	Configuration
Rounds	3
Batch Timeout	2sec
Preferred Max Bytes	512KB, 1MB
Transaction Send Rate(tps)	20, 40, and 60
Orderer and Size	Solo (Single Host) and 2Org with 1Peer each



(a) Throughput



(b) Avg. Latency

Figure 5.5: Impact of Batch Size on Throughput and Average Latency

Scenario 4: Performance Analysis using Distributed Storage System (IPFS). In this scenario, we analyse the performance using a distributed storage system by evaluating the time taken for storing and accessing medical records considering two metrics, i.e. upload time and download time. For analysis, the medical dataset with size 1GB was downloaded from Kaggle. Further, for scalability, 5

different datasets were derived from this with having sizes 20MB, 40MB, 60MB, 80MB and 100MB, respectively. These different datasets were uploaded to the distributed IPFS network by different entities of the proposed blockchain network and downloaded to the requester system. The upload time is observed during data is uploaded to the IPFS network. The download time is observed while fetching data from the IPFS using hash value available on the blockchain. The upload and download time is also affected by the internet speed. During the evaluation, the internet upload speed was 6.89Mbps, and the download speed was 13.1Mbps.

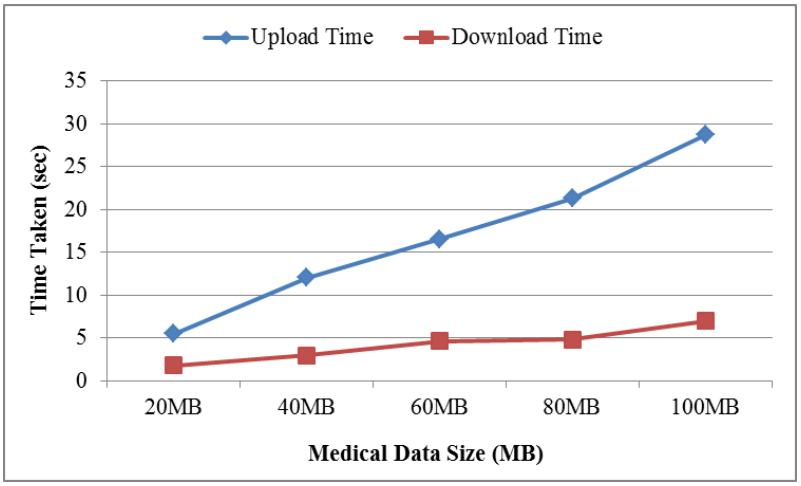


Figure 5.6: Average upload and download time using IPFS

Figure 5.6, shows the average time taken to upload and download records. It is observed that the proposed system takes less than 6 sec to upload 20MB of data and an average of 28 sec to upload 100MB of data. The maximum average download time is approximately 7 sec for 100MB of data.

From the above analysis, it is inferred that the overall proposed system takes approximately 280 sec and 256 sec in total to upload and download, respectively, for 100MB of data. Along with the upload and download time, this total time also includes blockchain processes such as network creation time, registration time and query processing time etc.

Scenario 5: Security Analysis of the designed Smart Contract. With smart contracts, there is always the possibility of unforeseen risks. Few critical ones

are unchecked exceptions, global variables, access to networks outside of the blockchain, and iteration on the map object itself etc. So a static study of SC is required prior to its real deployment. According to the authors,[124] static analysis tool checks, if the chaincode contains concurrency, unchecked exceptions, ledger operations dependent on global state, field declarations, or otherwise unclean input. These global variables could be utilised to exploit the ledger operations because they are only applicable to the peer the chaincode is put on. Due to its erroneous readset, if this peer experiences a crash or becomes unsynchronized with the remaining network, it will no longer be able to submit additional transactions. Conducting a static analysis of the chaincode guarantees that the implementation will remain impervious to such attacks. Chaincode Analyzer[125], an open-source tool, has been utilized to assess the potential risks associated with our built smart contracts. As demonstrated in Figure 5.7, the examination performed by this tool shows that our developed programmes are free of any faults.

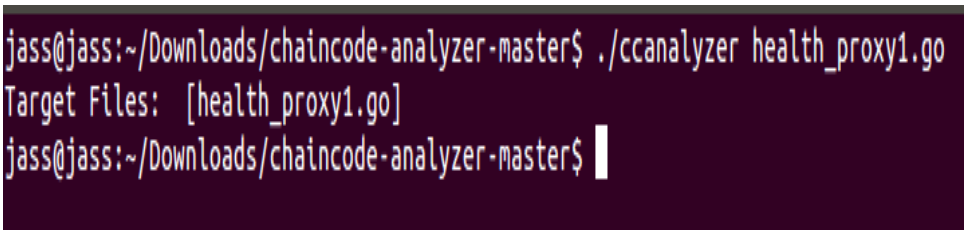


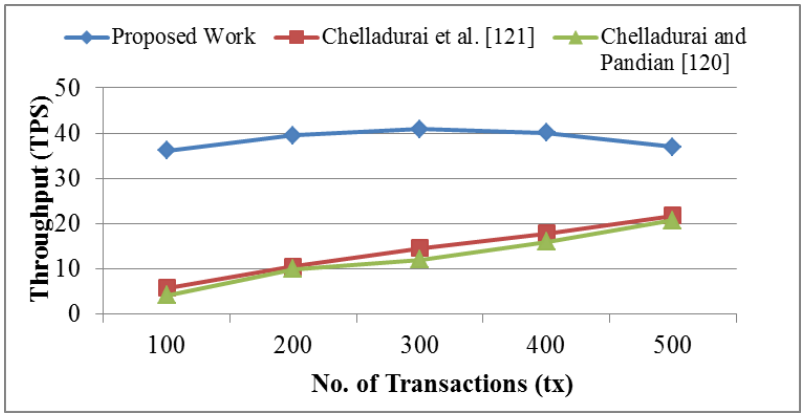
Figure 5.7: Chaincode Analyzer report of error-free code

Scenario 6: Performance Comparison of Proposed and Existing Systems. The performance comparison of related work[120,121] and proposed work is demonstrated in the form of graphs and table as shown in Figure 5.8, Table 5.6 and Table 5.7. For comparison of proposed and related existing work, only similar work (implementation and validation platform with similar assumptions and constraints) is considered. The feature-based comparison of the proposed and related work is given in Table 5.1 of Section 5.1.1.

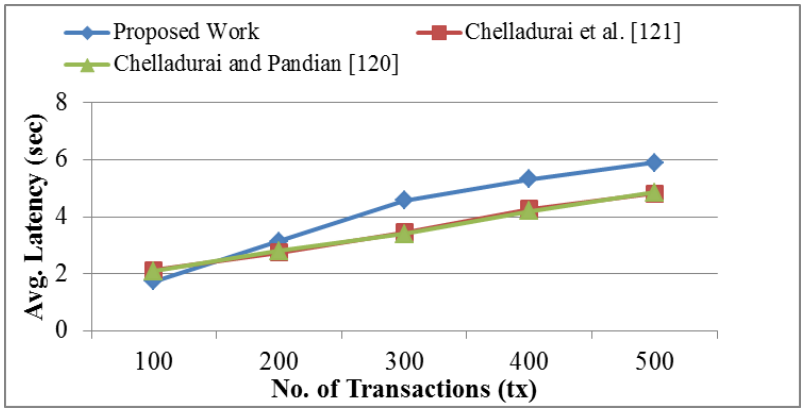
Figure 5.8 demonstrates the performance comparison of proposed and existing work by[120,121] based on throughput and average latency. Figure 5.8(a) and

Table 5.6: Performance Comparison of proposed and existing work in terms of throughput.

Reference → Transaction Group(s) ↓	**Proposed Work**	**Chelladurai and Pandian [120]**	**Chelladurai et al. [121]**
100	36.1	4.2	5.82
200	39.5	10	10.54
300	40.9	12	14.57
400	40.1	16	17.89
500	37	20.73	21.73



(a) Throughput



(b) Avg. Latency

Figure 5.8: Throughput and Average Latency comparison between proposed and existing work.

Table 5.6 depict that the proposed system outperforms the existing work in terms of throughput. As a result, from Figure 5.8(b) and Table 5.7, it is observed that the avg. transactional latency of the proposed system is marginally higher than the existing work.

Table 5.7: Performance Comparison of proposed and existing work in terms of Avg. Latency.

Reference → **Transaction Group(s) ↓**	**Proposed Work**	**Chelladurai and Pandian** [120]	**Chelladurai et al.** [121]
100	1.74	2.1	2.13
200	3.14	2.8	2.74
300	4.57	3.4	3.46
400	5.32	4.2	4.28
500	5.9	4.85	4.81

5.6 Chapter Summary

In this chapter, a permissioned Hyperledger blockchain based framework is proposed for privacy preservation and secured sharing of EHRs. The proposed framework is integrated with IB-PRE algorithm for secure sharing without exposing private keys and thus maintaining the privacy of records. IPFS is used as distributed storage for storing EHRs. It is a content-addressable storage that ensures the integrity of the content such that a slight modification in the stored EHR records results in a change in the obtained hash value. The encrypted records of patients are placed in IPFS to prevent malicious attacks or unauthorized access, while hash values corresponding to these records are placed in the blockchain distributed ledger. Finally, the *Hyperledger Caliper* tool is used to test the smart contracts and check the system's performance. Also, the performance comparison of existing related work[120,121] and proposed work is demonstrated. In addition to this, security analysis of the designed smart contracts and performance analysis using a distributed storage system *(IPFS)* are also performed. The results prove the feasibility and efficacy of the proposed system using throughput and latency (average) as the performance parameters.

Chapter 6

DUAL CHANNEL BLOCKCHAIN BASED FRAMEWORK FOR HEALTHCARE DATA

This chapter upgrades the framework by developing dual-channel blockchain architecture combined with two cryptographic algorithms, i.e. RSA and AES, to provide enhanced security and rapid retrieval of healthcare information. Further, the concept of private data collection is incorporated to securely store confidential patient information, guaranteeing privacy, security and limited access. Also, an ACL is defined for different users to implement access permissions, i.e., grant and revoke access to viewers while sharing information. The performance evaluation is performed by conducting experimental simulations, where critical performance indicators such as throughput and latency are measured across different transaction rates, channels, and rate controllers. Moreover, the proposed framework classifies smart contract functions into query and invoke/write transactions, enhancing the efficiency of data retrieval. Further, the functionality and security analysis of the proposed framework is discussed.

6.1 Motivation and Contributions

The research gaps identified emphasize the need to protect patient privacy while ensuring data accessibility for scientific research and medical purposes using scalable blockchain solutions. Researchers have attempted to address these challenges by improving consensus mechanisms, adding auxiliary channels, and implementing other approaches to make the system more secure, faster and scalable. To address and further improve the issues such as security, quick data retrieval and scalability, the following contributions are made:

- A consortium blockchain-based framework is proposed for securing and querying healthcare data, which considers multiple healthcare entities, including hospitals, doctors, patients, data analysts, and other Third-Party Alliances (TPAs).

- To enhance the security and confidentiality of transactions among various entities, multiple channels are created. Along with the regular communication among various entities, these channels support the Private Data Collection feature and thus define different access control policies for various entities.

- To provide secure storage, sharing, and scalability, two cryptographic algorithms, RSA for secure data transmission and AES for efficient storage, are implemented along with IPFS, an offline database, for scalability. IPFS stores actual EHRs, and consortium blockchain stores the respective EHR indexes.

- To grant access control while sharing data, the concept of an ACL is defined and utilized, which empowers the entities to secure the distributed storage and give data access to authorized users by providing an encrypted link.

Table 6.1 shows a characteristic comparison among the proposed work and the

relevant studies. The characteristics included are a patient-centric system that recognises the patient as the owner of their data, content-addressable off-chain storage to address scalability issues, private data collection with multi-channel capabilities to ensure data security and privacy and additional encryption techniques for securely sharing EHRs. In addition, it includes a performance study to assess the efficiency and viability of the proposed framework, as well as a vulnerability assessment of the blockchain system to verify its resilience to various types of attacks.

Table 6.1: Characteristics-based Comparison of Existing and Proposed Work

Characteristics → Reference(s)↓	**Patient Centric**	**Content addressable Distributed Storage**	**Private Data Collection Feature**	**Multi-Channel Implementation**	**Additional Cryptographic and Access Policy**	**Performance Assessment Performed**
[86]	Yes	No	No	No	No	Yes
[93]	Yes	No	No	No	Yes	No
[94]	Yes	Yes	No	No	Yes	Yes
[97]	Yes	No	Yes	Yes	No	Yes
[100]	Yes	No	No	No	No	No
[119]	Yes	Yes	No	No	No	Yes
Proposed System	Yes	Yes	Yes	Yes	Yes	Yes

6.2 Preliminaries and Background

The proposed system leverages advanced cryptographic algorithms, access control mechanisms, and decentralized storage solutions to improve the safety, confidentiality and effectiveness of managing healthcare data on a blockchain platform. This section provides the summary of the key technologies and concepts used in the proposed system, including RSA[126], AES[127], Access Control Lists (ACLs), and the InterPlanetary File System (IPFS).

1. **RSA (Rivest-Shamir-Adleman):** RSA is a popular asymmetric cryptographic method utilized extensively for secure data transmission[128]. The process includes two keys: a public key, which can be publicly shared, and a private key, which is securely protected. This algorithm mainly contains three algorithms: Key Generation, Encryption, and Decryption.

   Key Generation: Choose two large prime numbers, a and b, during the RSA key generation procedure. Then compute N as $(N = a\ddot{O}b)$. and the totient function $\phi(N)$ as $\phi(N)= (a-1)(b-1)$. An integer e is selected to be coprime with $\phi(N)$, such that the greatest common divisor of e and $\phi(N)$ is 1, where 1<e<$\phi(N)$. Determine the private key exponent d as the modular inverse of e modulo $\phi(N)$, denoted as $d \equiv e^{-}1mod\phi(N)$. The public key comprises the values (N,e), while the private key comprises the values (N,d).

   Encryption: Data M is encrypted using an encryption method using a public key to compute cipher text C as $C \equiv M^e mod(N)$.

   Decryption: The encrypted message as cipher text C is decrypted to obtain the original message using private key exponent d as $M \equiv C^d mod(N)$.

2. **AES (Advanced Encryption Standard):** AES is a symmetric encryption method that is used to ensure data security[127]. This method encrypts data with a secret key (128/192/258 bits) and divides it into fixed-size blocks (128 bits). AES is known for its exceptional speed and robust security and is, therefore, ideal for encrypting large amounts of data. AES is used to encrypt health data before it is placed on the blockchain to ensure its confidentiality[129]. Authorized parties use the same AES key to decrypt the data. The key expansion method is used to derive round keys from output keys that are the result of various operations such as substitution, rotation and XOR operations.

   Encryption and Decryption: AES encrypts data by applying a series of

substitution, permutation and mixing operations to a fixed-size block of data across multiple rounds. During decryption, these operations are reversed to restore the original data.

3. **Access Control Lists (ACLs):** These lists are the security mechanism that defines the permissions granted to individuals or system processes to access objects. They also define the activities that can be carried out with certain objects. ACLs are essential for maintaining data security and protecting privacy. In the proposed system, patients can allow specific individuals, such as physicians, insurance companies or data analysts, to access their health data in read/write and view/create mode.

6.3 Proposed Framework and Methodology

This section discusses the proposed framework as shown in Figure 6.1, with the detailed steps followed that can be used for securing and querying healthcare data with hospitals, doctors, patients, researchers, insurers, laboratories, pharmacies, and other stakeholders. The proposed framework follows the proposed methodology consisting of four phases, each with detailed steps, as illustrated in Figure 6.2. Such as the network designing step, data model and chaincode creation step, setting up of infrastructure and identity registration step, data collection and storage step, access control for sharing, querying, testing, and performance measure step. The health records are encrypted using a combination of RSA and AES cryptographic algorithms along with the patient's public key and symmetric keys. This ensures data confidentiality and maintains integrity. The encrypted EHR is then placed on IPFS. The security and authenticity of the framework are enhanced by designing smart contracts, access policies and a private data collection feature.

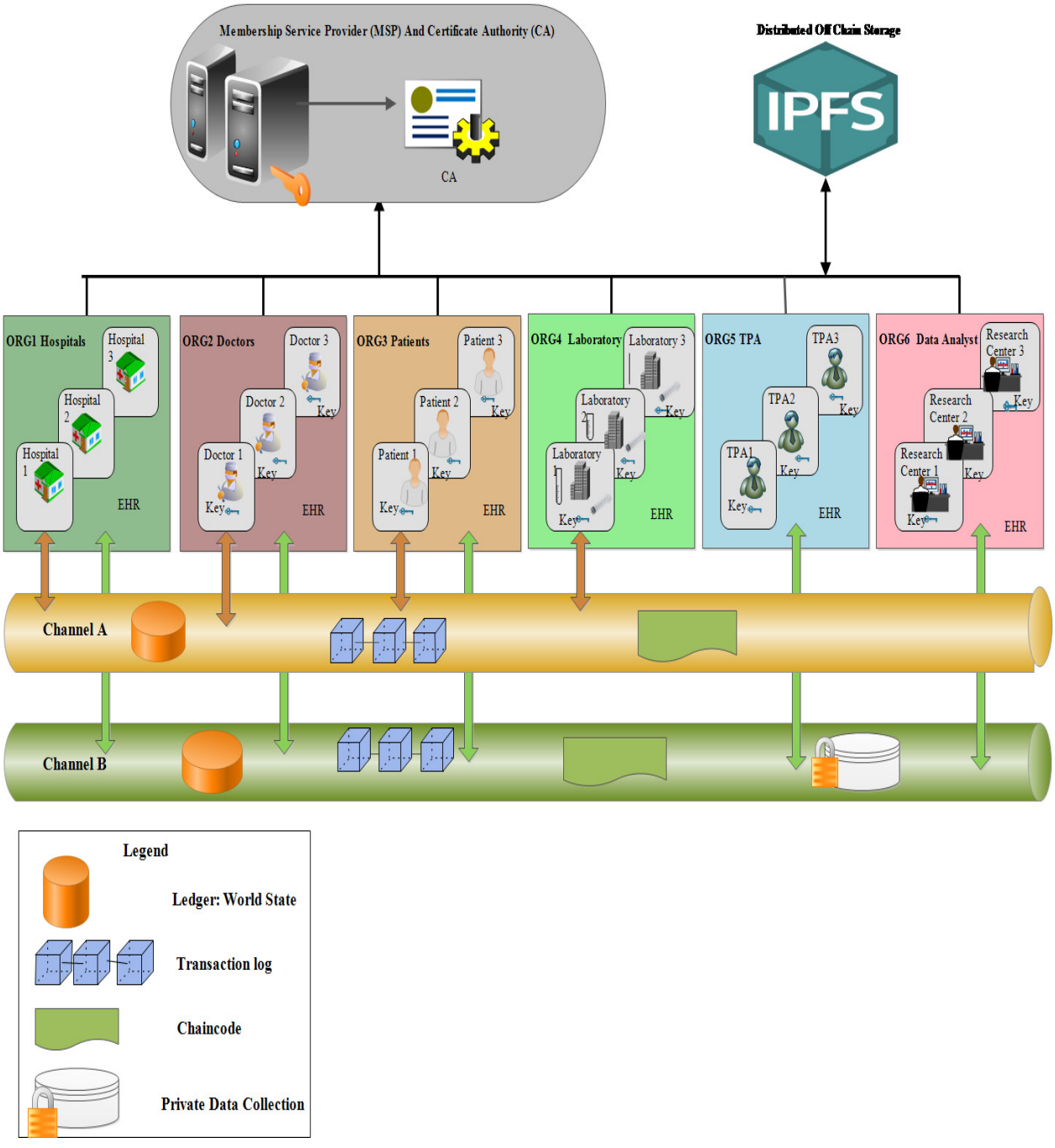


Figure 6.1: Architecture of the Proposed Framework

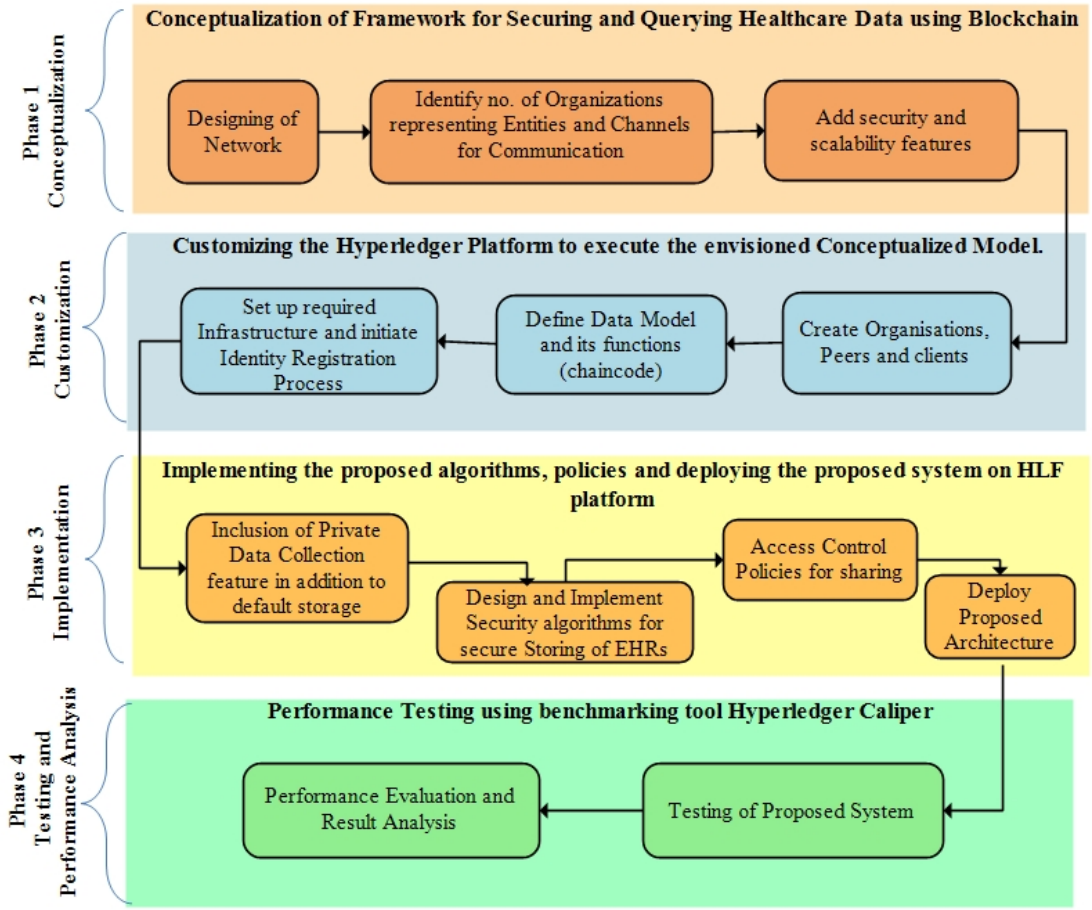


Figure 6.2: Methodology followed for designing and developing proposed framework

Figure 6.2 presents the methodology followed for designing and developing the proposed framework, which consists of four phases, as discussed below.

Phase 1: Conceptualization of the Proposed Framework

This work focuses on designing a patient-centric system for securely sharing and querying healthcare data among various entities as organizations in a decentralized network. A permission blockchain regulates access to the network and ensures that only authorized individuals are permitted to join. Each participant in the network can generate transactions, store, exchange, access, and alter medical records based on the permissions allowed under a system of regulations that functions within the decentralized network. In order to implement

these regulations, the blockchain must possess smart contracts. Every participant must be a member of at least one of these organizations and can hold multiple user roles such as regulatory authority, doctor, patient, etc.

The system's design effectively safeguards user's confidential data through encryption and decryption methods, preventing unauthorized access both while at rest and in transit. In addition, the concept of dual chain and private data collection is implemented to address the scalability issue and manage EHR records efficiently and privately within the organizations, as shown in Figure 6.1. Off-chain storage solutions are utilized to lessen the burden on the main blockchain and enhace the overall system performance.

Steps 1 and 2: Designing of Network and Selection of Organizations: The first important step in the proposed architecture is the planning and designing of the network. The planning involves the number of organizations that will participate in the network, the peers size and orderers for each organization, and the communication channels between the organizations. In the proposed work, the network consists of multiple organizations such as hospitals, doctors, patients and so on, each with their own peers, endorsing peers, certificate authority and CouchDB as a state database. The various organizations and the communication channels that are considered in the proposed network and their functionality are discussed below.

A. Organizations

(a) Hospitals: Hospitals are the primary healthcare providers in the network. They maintain the medical records of patients, including diagnoses, treatment plans, lab results, and insurance information. Hospitals have access to a private channel (Channel B) that includes only other hospitals and the insurance agents they work with. This private channel allows hospitals to securely and confidentially

communicate with other healthcare providers and insurers to manage patient care and insurance claims.

(b) Doctors: This organization employs doctors who treat patients at hospitals. Doctors are responsible for providing medical treatment to patients in the hospitals. They have access to the same private channel (Channel B) as hospitals, allowing them to access patient records and collaborate with other healthcare providers and insurers with the patient's consent.

(c) Patients: This organization represents the patients who own their data. Patients are the individuals receiving medical care in the network. They have a unique ID on the network, which is used to access their medical records and insurance information. They can communicate with doctors and hospitals through a separate channel (Channel A), which is only accessible to patients and the hospitals, doctors, and lab technicians they are working with. This allows for secure and private communication between patients and their healthcare providers.

(d) Lab Technicians: This organization performs lab tests on patients' samples. Lab technicians are responsible for conducting medical tests and providing results to hospitals, doctors, and patients. They have their own node on the network and access to the private channel (Channel A) that includes hospitals and insurance agents. This allows for secure and confidential communication between labs, hospitals, and insurers regarding test results and insurance coverage.

(e) Researcher: This organization conducts medical research on patient data. Researchers are responsible for conducting medical research and analyzing patient data to improve healthcare outcomes. They have their own node on the network and access to a separate channel (Channel

B) that includes only them and the hospitals they work with. This allows for secure and private communication between researchers and hospitals regarding medical research and patient data.

(f) TPA: This organization manages insurance claims and payments. Insurance agents are responsible for managing insurance coverage and patient claims processing. They have their own node on the network and access to the private channel (Channel B) that includes hospitals and lab technicians. This allows for secure and confidential communication between insurers and healthcare providers regarding insurance coverage and claims processing.

B. Channels

Multiple channels can be created within the network to separate the data access and processing for different healthcare stakeholders. In the proposed work, dual channel architecture is designed, i.e., the Healthcare Channel and the Research Channel.

(a) Healthcare Channel (Channel A): This channel is used by hospitals, doctors, lab technicians, TPA and patient organizations to share patient data and information related to medical services with other intended, authorized stakeholders.

(b) Research Channel (Channel B): This channel is used by the Researcher organization to access and analyze the anonymized healthcare data for research purposes.

Step 3: Add Security and Scalability Features: To guarantee that only authorized entities can access and edit data, a permissioned blockchain system is employed. This system allows participants to join the network

only after completing registration and obtaining authorization. In addition to this, data encryption techniques i.e RSA and AES are employed to Encrypt healthcare data at rest and in transit to safeguard it from unauthorized access. Furthermore, smart contracts are utilized to implement access control measures, ensuring that only authorized individuals can access certain data. This ensures that sensitive information is only accessible to those with the right permissions. The immutability feature of blockchain provides an Immutable Audit Trail, such that every transaction or access to healthcare data is recorded, providing transparency and accountability. IPFS off-chain distributed storage is used to address scalability issues.

Phase 2: Customization: Customizing the Hyperledger Platform (HLF) to execute the envisioned Conceptualized Model

Step 1: Creating peers, clients and orderers: For implementing our proposed concept, the HLF platform is selected to create a consortium blockchain network where the regulatory authority regulates and registers various organizations. Further, one peer from each organization is included in the proposed work. Peers are responsible for managing ledgers and smart contracts. They also manage transaction proposals and endorsements, thus categorizing them as endorsing and committing peers. Also, there are clients and orders in the network. Clients initiate transactions by creating proposals and submitting them to endorsing peers. Endorsing peers simulate and endorse transactions, and then they are sent to the orderers for consensus on the order. Once consensus is reached, the committing peers commit the transactions to the ledger. Thus, HLF architecture offers organizations a decentralized and safe way to communicate within a permissioned blockchain network, ensuring immutability and privacy.

Step 2: Data model and chaincode designing: The next step is to define the data model and chaincode for the healthcare network. The data model defines the structure of the healthcare data, such as patient records, lab reports, prescription history, and insurance claims, and how it is stored and accessed. The chaincode defines the application logic and regulations for the network, such as how the healthcare data can be queried, updated, and shared between different stakeholders. The language used for chaincode implementation is Golang. Figure 6.3 Illustrates the data models in the form of structures used to develop chain codes with access policies.

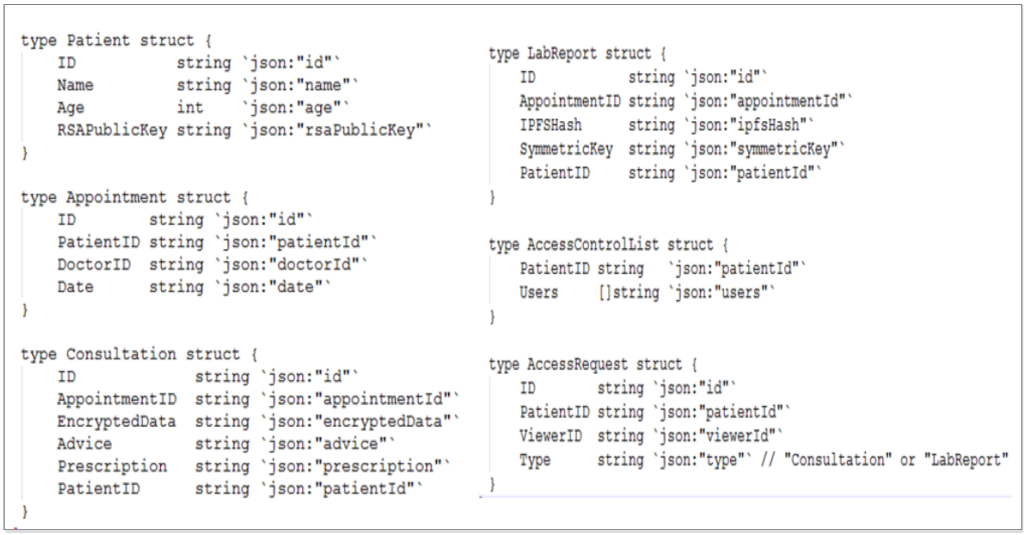


Figure 6.3: Structures for various entities and access policies

Step 3: Setting up Infrastructure and Identity Registration Process: Once the network, data model and chaincode is designed, the next step is to set up the necessary infrastructure to run the network. This includes setting up the virtual servers for the peers, orderers, and other network components, installing the required software (Docker, Go, and Hyperledger Fabric), and configuring the network components to communicate with each other. To secure the healthcare network, we implement the registration process for the stakeholders. This involves setting up a Certificate Authority (CA) to issue digital certificates along with public/private keys for each stakeholder using a Membership Service Provider (MSP) [130].

Phase 3: Implementation

This phase involves implementing the suggested algorithms, policies, and other features and then deploying the suggested framework.

Step 1: Private Data Collection in Addition to Default Storage: The main objective of this step is to offer secure storage and controlled access to sensitive healthcare information, ensuring that such records are only accessible to designated organizations within the network. In this work, implementation involves the use of collections configuration files to define private collections, as shown in Figure 6.4

```
[
  {
    "name": "privateCollection",
    "policy": "OR('Org1MSP.member', 'Org2MSP.member','Org6MSP.member')",
    "requiredPeerCount": 1,
    "maxPeerCount": 3,
    "blockToLive": 1000000,
    "memberOnlyRead": true,
    "memberOnlyWrite": true,
    "endorsementPolicy": {
     "signaturePolicy": "OR('Org1MSP.member','Org2MSP.member','Org6MSP.member')"
   }
  }
]
```

Figure 6.4: Configuration.json file for implementing private collection feature

Step 2: Designing and Implementing Security Algorithms for Secure Storing and Sharing of EHRs: In this step, different security algorithms such as RSA (asymmetric) and AES (symmetric) are designed, implemented and integrated with the system to ensure EHRs are stored securely within the blockchain network to protect them from unauthorized access and tampering, ensuring data integrity and confidentiality. Here's an overview of how these algorithms are utilized and implemented in the proposed system.

The RSA algorithm comprises three algorithms for secure key exchange and ensuring that only allowed entities can access sensitive data. Algorithm 11 is used to create RSA keys (public and private) for various entities

during registration on the network to encrypt and decrypt data securely. Algorithm 12 and Algorithm 13 are used to encrypt sensitive information using public keys before its storage on the blockchain and corresponding private keys to decrypt and access the data by authorized entities, respectively.

Algorithm 11 RSAKeyGen()

1: **function** RSAKEYGEN()
2: $\langle pubKey, privKey \rangle \leftarrow$ RSA.GENERATEKEYPAIR
3: **return** $\langle pubKey, privKey \rangle$
4: **end function**

Algorithm 12 RSAEncryption()

Input: EHR data such as advice, prescriptions, notes
Output: Ciphertext as encryptedData

1: **function** RSAENCRYPTION()($data$, $pubKey$)
2: $encryptedData \leftarrow$ RSA.ENCRYPT$(data, pubKey)$
3: **return** $encryptedData$
4: **end function**

Algorithm 13 RSA Decryption

Input: encryptedData, privKey **Output:** Original EHR

1: **function** RSADECRYPTION(encryptedData, privKey)
2: $EHR \leftarrow$ RSA.DECRYPT$(encryptedData, privKey)$
3: **return** EHR
4: **end function**

Similarly, in AES encryption, Algorithm 14, Algorithm 15, Algorithm 16 is used for fast and secure data encryption and decryption, ensuring privacy and security within the blockchain. Here, the same key is used for the encryption and decryption of the EHR. This is utilized by the Lab technicians while sharing lab reports with patients and doctors.

Algorithm 14 AESKeyGen()

1: **function** AESKEYGEN
2: $aesKey \leftarrow$ AES.GENERATEKEY
3: **return** $aesKey$
4: **end function**

Algorithm 15 AESEncryption()

Input: Lab Reports as data i.e Ldata
Output: Ciphertext as LabencData

1: **function** AESENCRYPTION()($Ldata$, $aesKey$)
2: $LabencData \leftarrow$ AES.ENCRYPT($Ldata, aesKey$)
3: **return** $LabencData$
4: **end function**

Algorithm 16 AESDecryption()

Input: Ciphertext i.e LabencData
Output: LData

1: **function** AESDECRYPTION()($LabencData$, $aesKey$)
2: $data \leftarrow$ AES.DECRYPT($LabencData, aesKey$)
3: **return** $Ldata$
4: **end function**

Step 3: Access Control Policies for Sharing Records: The access control policies in the proposed system ensure that healthcare data is securely shared with authorized entities, such as requesters, data analysts, insurance companies, and other stakeholders. These policies are implemented through a combination of smart contracts (chaincode) and the underlying Hyperledger Fabric infrastructure. The working and functionality of implemented ACLs are discussed below in the form of Algorithms. Algorithm 17 and Algorithm 18 are used to authorize or withdraw authorization from doctors to create EHRs and upload them to the ledger, where DoctorID is passed as an argument for userID. During the grant process, the user is added to the patient's ACL list, while during revoke, the user is removed from the ACL list.

Request Access: Entities such as insurance companies or data analysts can request access to patient data by invoking the RequestAccess() function described in Algorithm 19. Further, Patients approve requests by invoking Algorithm 20.

The proposed framework implements robust access control policies, i.e., using (ACLs) to ensure the secure sharing and access of healthcare data.

Algorithm 17 GrantAccess

```
1: function GRANTACCESS(patientID, userID)
2:     aclAsBytes ← GETSTATE(patientID + "_acl")
3:     if aclAsBytes = NULL then
4:         acl ← {PatientID : patientID, Users : [userID]}
5:     else
6:         acl ← JSON.UNMARSHAL(aclAsBytes)
7:         acl.Users.append(userID)
8:     end if
9:     aclAsBytes ← JSON.MARSHAL(acl)
10:    PUTSTATE(patientID + "_acl", aclAsBytes)
11:    return "Access granted successfully"
12: end function
```

Algorithm 18 RevokeAccess

```
1: function REVOKEACCESS(patientID, userID)
2:     aclAsBytes ← GETSTATE(patientID + "_acl")
3:     if aclAsBytes = NULL then
4:         return "ACL does not exist for patient"
5:     end if
6:     acl ← JSON.UNMARSHAL(aclAsBytes)
7:     acl.Users.remove(userID)
8:     aclAsBytes ← JSON.MARSHAL(acl)
9:     PUTSTATE(patientID + "_acl", aclAsBytes)
10:    return "Access revoked successfully"
11: end function
```

Algorithm 19 RequestAccess()

```
1: function REQUESTACCESS()(ctx, requestID, patientID, requesterID,
   accessType)
2:     accessRequest ← {RequestID : requestID, PatientID : patientID, RequesterID :
   requesterID, AccessType : accessType}
3:     requestAsBytes ← JSON.MARSHAL(accessRequest)
4:     CTX.GETSTUB().PUTSTATE(requestID, requestAsBytes)
5:     return "Access request recorded successfully"
6: end function
```

Algorithm 20 ApproveAccessRequest

```
1: function APPROVEACCESSREQUEST(ctx, requestID)
2:     accessRequestAsBytes ← CTX.GETSTUB().GETSTATE(requestID)
3:     accessRequest ← JSON.UNMARSHAL(accessRequestAsBytes)
4:     accessRequest.Granted ← true
5:     updatedRequestAsBytes ← JSON.MARSHAL(accessRequest)
6:     CTX.GETSTUB().PUTSTATE(requestID, updatedRequestAsBytes)
7:     return "Access request approved successfully"
8: end function
```

These policies allow patients to grant and approve access, log access requests, and verify permissions before providing data access. This approach ensures data security, patient privacy, regulatory compliance, data integrity, and system transparency.

Step 4: Deploying proposed version: After setting up the required infrastructure, creating organizations, peers, and clients, and designing data models and chain code, the proposed system is deployed for testing on the HLF platform by following the below steps:

Installing and Approving Chaincode: This includes preparing the chain code for installation by packaging it into a tar.gz file, followed by installing the chain code on all peer nodes in the network. Finally, Each organization must approve the chain code before it can be committed to the channel.

Committing Chaincode Definition: Commit the chaincode definition to the channel after all organizations have approved it.

Invoking Chaincode Initialization: Initialize the chain code to set up any necessary data or state to ensure the initialization transaction is successfully committed.

The complete working of the proposed approach leveraging various designed algorithms and policies is discussed as a communication diagram. Figure 6.5 illustrates the interaction among multiple entities after all the entities are successfully registered in the network, as discussed below:

1: After successfully registering in the network by invoking the *RegisterPatient( )* function as defined in Algorithm 21, the patient goes to the hospital to receive treatment and schedule an appointment. The hospital entity makes an appointment by invoking the *MakeAppointment( )* function as Algorithm 22 and also notifies the doctor about the patient's appointment.

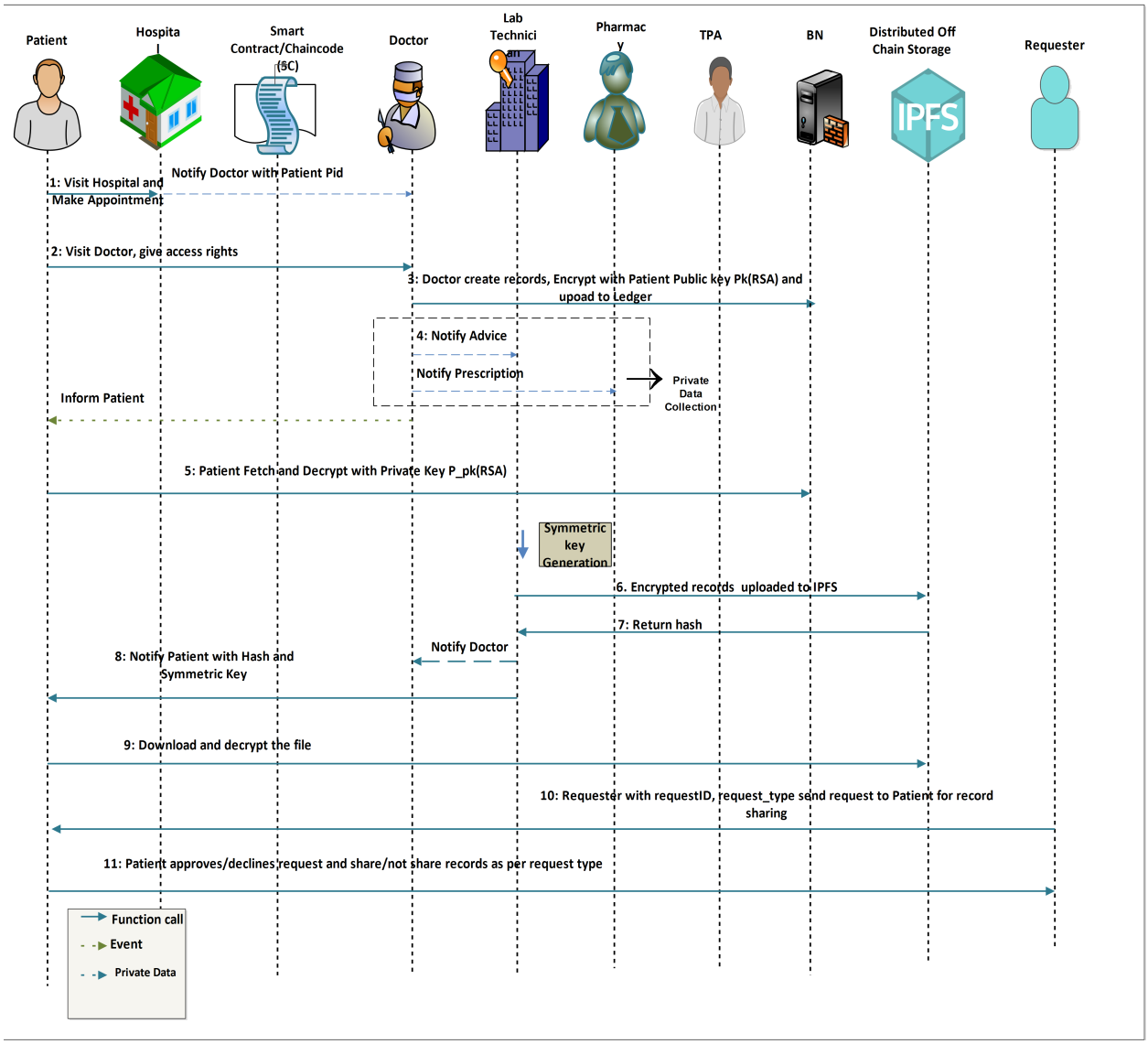


Figure 6.5: Working of the Proposed Framework

Algorithm 21 RegisterPatient()

```
1: function RegisterPatient()(patientID, name, age)
2:     ⟨pubKey, privKey⟩ ← RSAKeyGen
3:     patient ← {ID : patientID, Name : name, Age : age, RSAPublicKey : pubKey}
4:     patientAsBytes ← JSON.Marshal(patient)
5:     PutState(patientID, patientAsBytes)
6:     return "Patient registered successfully"
7: end function
```

2-3: The patient visits the doctor for consultation and is granted access to create and view their EHRs as discussed in Algorithm 17 and Algorithm 18. Then, the doctor, after treatment, creates new EHRs, encrypts them with the patient's public Key, and then uploads the records to the ledger.

Algorithm 22 MakeAppointment()

```
1: function MAKEAPPOINTMENT()(appointmentID, patientID, doctorID, date)
2:     patientAsBytes ← GETSTATE(patientID)
3:     if patientAsBytes = NULL then
4:         return "Patient does not exist"
5:     end if
6:     appointment ← {ID : appointmentID, PatientID : patientID, DoctorID :
   doctorID, Date : date}
7:     appointmentAsBytes ← JSON.MARSHAL(appointment)
8:     PUTSTATE(appointmentID, appointmentAsBytes)
9:     return "Appointment made successfully"
10: end function
```

4: After uploading to the ledger, the doctor also informs Lab Technicians about advice parameters for conducting medical tests and Pharmacists about prescriptions in a secure way using a private data collection feature as described in Algorithm 23.

Algorithm 23 RecordConsultation()

```
1: function RECORDCONSULTATION()(consultationID, appointmentID,
   consultationData, advice, prescription)
2:     appointmentAsBytes ← GETSTATE(appointmentID)
3:     if appointmentAsBytes = NULL then
4:         return "Appointment does not exist"
5:     end if
6:     appointment ← JSON.UNMARSHAL(appointmentAsBytes)
7:     patientAsBytes ← GETSTATE(appointment.PatientID)
8:     if patientAsBytes = NULL then
9:         return "Patient does not exist"
10:    end if
11:    patient ← JSON.UNMARSHAL(patientAsBytes)
12:    encryptedData ← ENCRYPTWITHRSA(patient.RSAPublicKey, consultationData)
13:    consultationRecord ← {ID : consultationID, AppointmentID :
   appointmentID, EncryptedData : encryptedData, Advice : advice, Prescription :
   prescription, PatientID : appointment.PatientID}
14:    consultationRecordAsBytes ← JSON.MARSHAL(consultationRecord)
15:    PUTPRIVATEDATA("privateCollection", consultationID, consultationRecordAsBytes)
16:    SETEVENT("NotifyPatient", "Consultation record updated for appointment "+
   appointmentID)
17:    SETEVENT("NotifyLabTechnician", "New advice and prescription for appointment "+
   appointmentID)
18:    return "Consultation recorded successfully"
19: end function
```

5: After getting a notification from a doctor, the patient fetches the records from the ledger calling *GetConsultationRecord()* function defined in Algorithm 24.

Algorithm 24 GetConsultationRecord()

```
1: function GetConsultationRecord()(consultationID, userID)
2:     consultationRecordAsBytes ← GetPrivateData("privateCollection", consultationID)
3:     if consultationRecordAsBytes = NULL then
4:         return "Consultation record does not exist"
5:     end if
6:     consultationRecord ← JSON.Unmarshal(consultationRecordAsBytes)
7:     if checkAccess(consultationRecord.PatientID, userID) = FALSE then
8:         return "Access denied"
9:     end if
10:    return consultationRecord
11: end function
```

6-9: The Lab technician, after performing medical tests, generates a symmetric key, encrypts the patient's medical reports with a symmetric key and adds them to IPFS. After getting the hash of the corresponding record, he notifies the doctor about the same. Also, inform the patient with a symmetric key and hash value. Ultimately, the patient fetches the data from IPFs and decrypts locally to see the reports calling Algorithm 25 and Algorithm 26.

Algorithm 25 AddLabReport()

```
1: function AddLabReport()(reportID, appointmentID, labReportData)
2:     appointmentAsBytes ← GetState(appointmentID)
3:     if appointmentAsBytes = NULL then
4:         return "Appointment does not exist"
5:     end if
6:     appointment ← JSON.Unmarshal(appointmentAsBytes)
7:     ⟨symmetricKey, encryptedData⟩ ← EncryptWithSymmetricKey(labReportData)
8:     ipfsHash ← StoreInIPFS(encryptedData)
9:     SaveSymmetricKeyToFile(appointment.PatientID, symmetricKey)
10:    labReport ← {ID : reportID, AppointmentID : appointmentID, IPFSHash :
       ipfsHash, SymmetricKey : symmetricKey, PatientID : appointment.PatientID}
11:    labReportAsBytes ← JSON.Marshal(labReport)
12:    PutPrivateData("privateCollection", reportID, labReportAsBytes)
13:    SetEvent("NotifyPatient","Lab report added for appointment " +
       appointmentID + " IPFS Hash: " + ipfsHash)
14:    SetEvent("NotifyDoctor","Lab report added for appointment " +
       appointmentID + " IPFS Hash: " + ipfsHash)
15:    return "Lab report added successfully"
16: end function
```

Algorithm 26 GetLabReport()

```
1: function GETLABREPORT()(reportID, userID)
2:     labReportAsBytes ← GETPRIVATEDATA("privateCollection", reportID)
3:     if labReportAsBytes = NULL then
4:         return "Lab report does not exist"
5:     end if
6:     labReport ← JSON.UNMARSHAL(labReportAsBytes)
7:     if CHECKACCESS(labReport.PatientID, userID) = FALSE then
8:         return "Access denied"
9:     end if
10:    return labReport
11: end function
```

Step 10-11: Any requestor such as a doctor, lab technician, researcher or TPA requests a patient for their data, and then the patient, after access checking in ACL, grants or revokes access utilising Algorithm 19 and Algorithm 20.

Phase 4: Testing and Performance Evaluation of Proposed Framework

In this phase, the testing and performance analysis of the proposed blockchain framework is performed. Initially, the functionality of the system is assessed by examining various test cases, followed by an experimental analysis of performance considering throughput and latency. Finally, the security analysis is done to assess whether the system is resistant to various attacks.

a) Evaluation of System Functionality

The proposed framework consists of multiple entities, each entity executing distinct functions as specified in the previously mentioned algorithms. This section examines and demonstrates the functionality of each entity using visual representations as Figure 6.6-Figure 6.8.

- Test Case 1: Register a new patient with the patient ID "patient1" and basic information such as name, age and an RSA public key. Figure 6.6 illustrates the data stored in the world state, i.e., CouchDB, after successful registration.

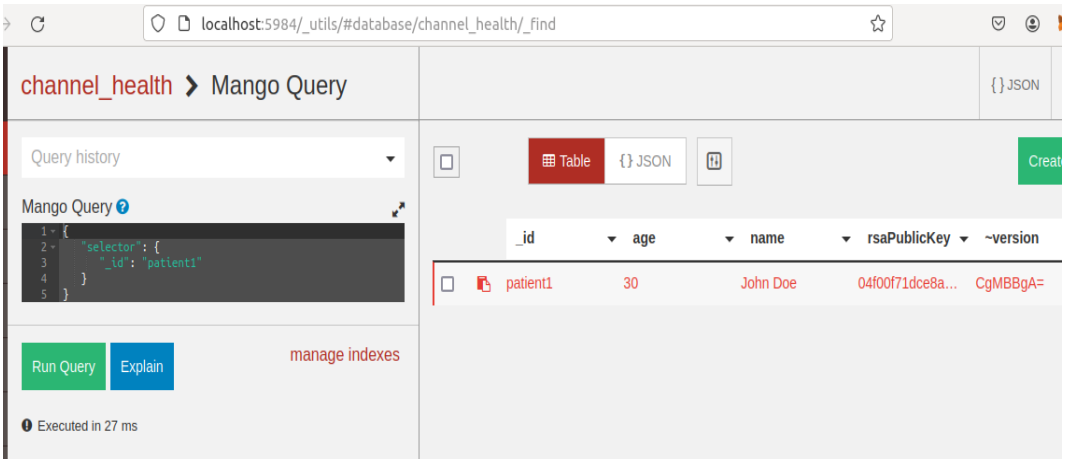


Figure 6.6: Successful Registration of a new Patient

- Test Case 2: Make an appointment for a registered patient, “patient1”, with a specific doctor using the MakeAppointment function. The appointment details are recorded in the blockchain to ensure immutability and traceability and to verify the patient’s existence before the appointment is made as shown in Figure 6.7.

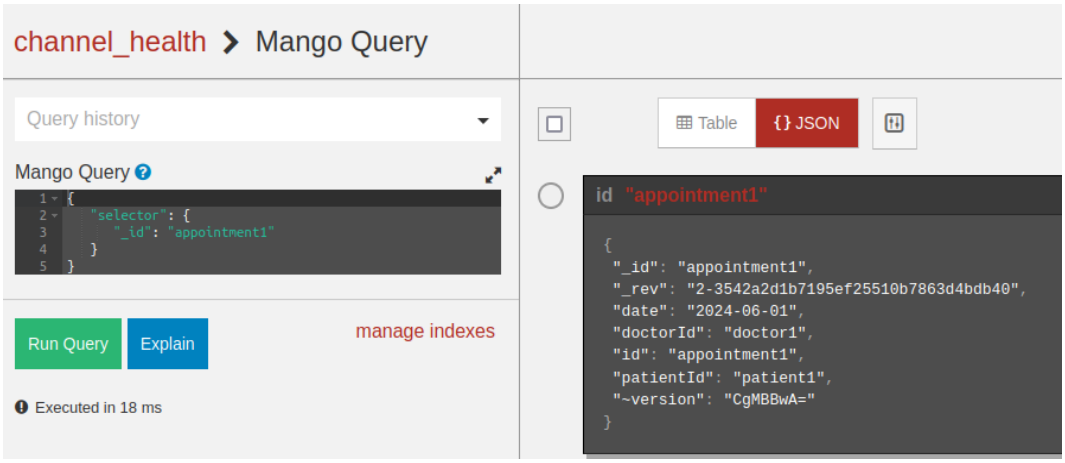


Figure 6.7: Successful Make Appointment

- Test Case 3 and 4: Consult the doctor, create a record and notify the lab technician along with adding a report by the lab technician using the Private Data Collection feature shown in Figure 6.8.

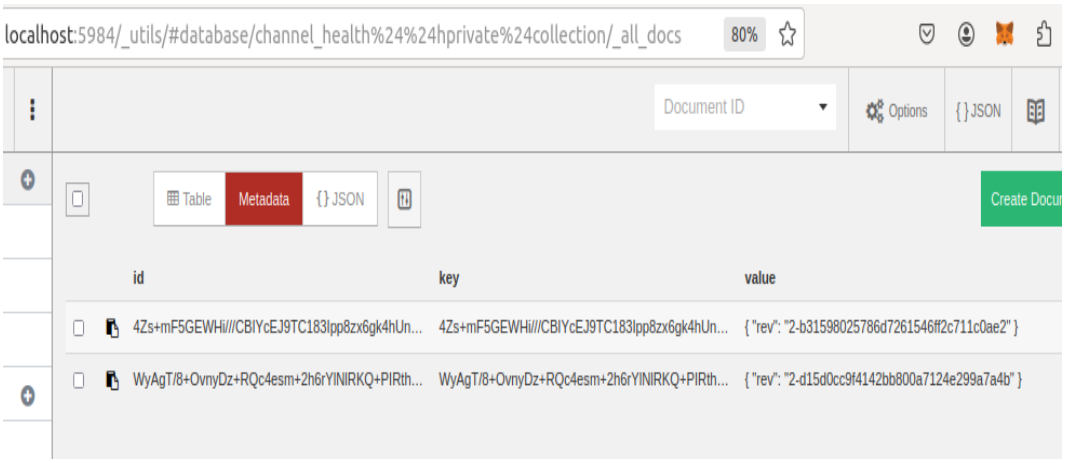


Figure 6.8: Showcase of Private Data Collection Storage

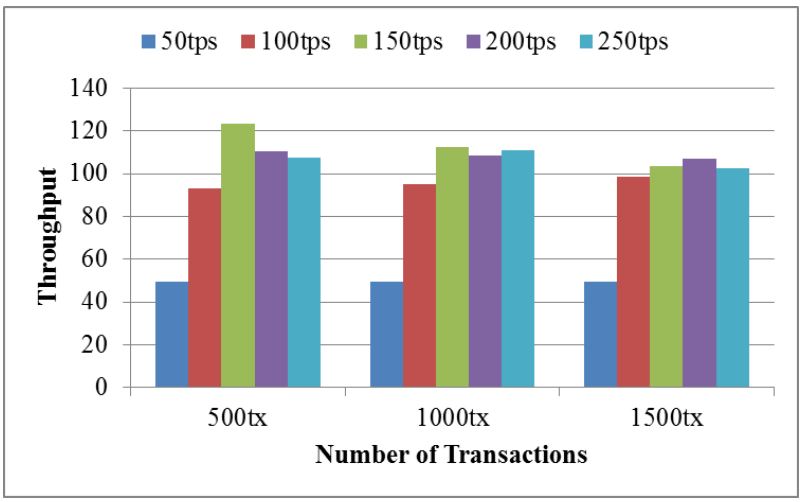
b) Evaluating System Performance

This section represents the experimental evaluation of the proposed Blockchain Network (BN). The performance of the deployed BN is measured using the Hyperledger Caliper benchmarking tool. The number of organizations, peers, channels, block size, transaction type, transaction size, and so forth all affect the system's performance. The performance is measured in terms of throughput and latency. The system configuration that is employed to implement and assess the proposed architecture is 12th Gen Intel(R) Core(TM) i7-12700 2.10 GHz processor, 16.0 GB of memory, Operating system Ubuntu Linux 18.04.1LTS, Docker Engine Version 24.0.2, Docker Compose Version 1.26.2 and Golang Language Opt for a smart contract, and Hyperledger Fabric v2.2.

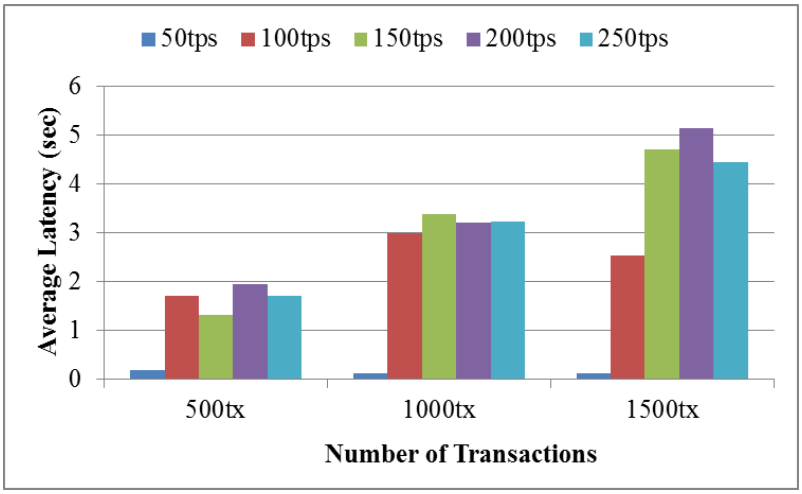
Different scenarios are considered to understand, evaluate, and analyse the proposed system. In these scenarios, the transaction size, rate, number of channels or type of rate controller varies. The performance is measured as transaction throughput in terms of transactions per seconds (TPS) and latency as seconds (sec).

Scenario 1: Primary Evaluation: In this evaluation, the impact of variation in transactions size (tx) and transaction rate (tps) is examined and various performance parameters such as throughput and latency are measured. The

evaluation considers the number of transactions as 500, 1000, and 1500 executed at the transaction rate of 50,100,150, 200 and 250 tps each. The effect of change of both transaction rate and size on throughput and average latency is shown in Figure 6.9.



(a) Transaction Rate vs. Throughput



(b) Transaction Rate vs. Average Latency (sec)

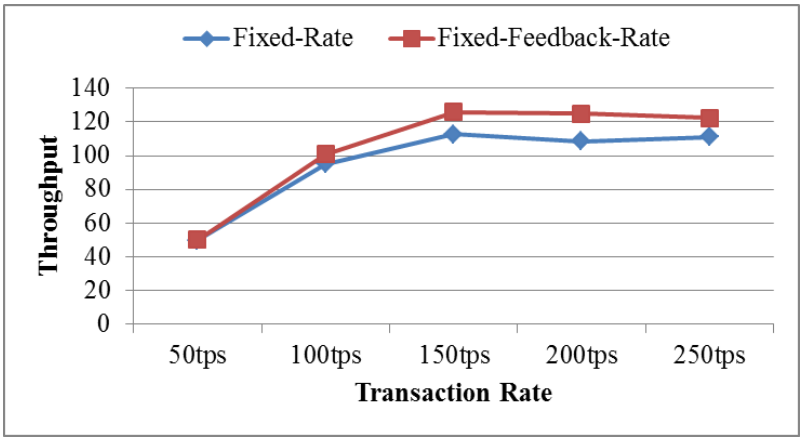
Figure 6.9: Effect of change of transaction rate (tps) on throughput (TPS) and latency (sec)

Figure 6.9a depicts that at a low transaction rate (tps), throughput remains consistent at around 49.8 across all transaction sizes (500tx,1000tx,1500tx). It shows that with the increase in tps rate, initially, the throughput increases at a certain point, i.e. peak value 123 at 150 tps for 500 transactions and above that system starts degrading.

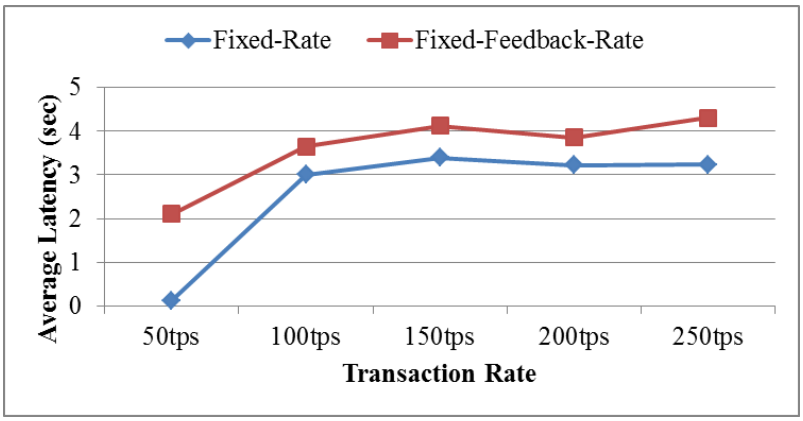
Similarly, Figure 6.9b depicts the average latency of the system; at a low transaction rate (tps), latency is minimal across all transaction sizes (500tx, 1000tx, 1500tx). Thus, at a low tps value, the system performs well with minimal delay. It is clearly observed that at higher transaction rates, the latency increases, particularly for larger transaction counts.

Scenario 2: Effect of different rate controllers on performance: The rate controllers are the key factors that affect performance. These are defined as the rates at which transactions are transmitted to the blockchain, such as fixed rate, fixed feedback rate, etc. The effect of these controllers on the proposed framework is examined. For analyses, two rate controllers, fixed rate and fixed feedback rate, are considered for 1000 transaction size at various transaction rates (50,100,150,200,250). The fixed-rate controller transmits input transactions at a predetermined TPS interval, while the fixed feedback rate controller enhances this. If the number of unfinished transactions exceeds the specified time, the system temporarily pauses to send inputs by sleeping for a period of time. Figure 6.10 Illustrates that with the increase in transaction rate, the throughput increases in the fixed-feedback rate controller as compared to a fixed rate. Similarly, with an increase in throughput, the latency also increases.

Scenario 3: Performance based on channels: The number of channels also affects the performance of the system In this scenario, the comparison based on channels such as single channel and dual channel is examined. For this comparison the number of transactions 1000 with fixed-rate controller type is considered. Figure 6.11 shows the effect of the number of channels on the performance. It is observed that the throughput increases and latency decreases in dual-channel networks as compared to single-channel. It is observed that throughput increases from 25 to 112.7 TPS at a transaction rate of 150 in dual channel, making the system more efficient such that it can handle more load efficiently as compared to a single channel. Similarly, the latency drops from 20.33 to 3.33 seconds, which



(a) Throughput



(b) Average Latency (sec)

Figure 6.10: Impact of different rate controllers on throughput and average latency (sec)

improves the system performance significantly.

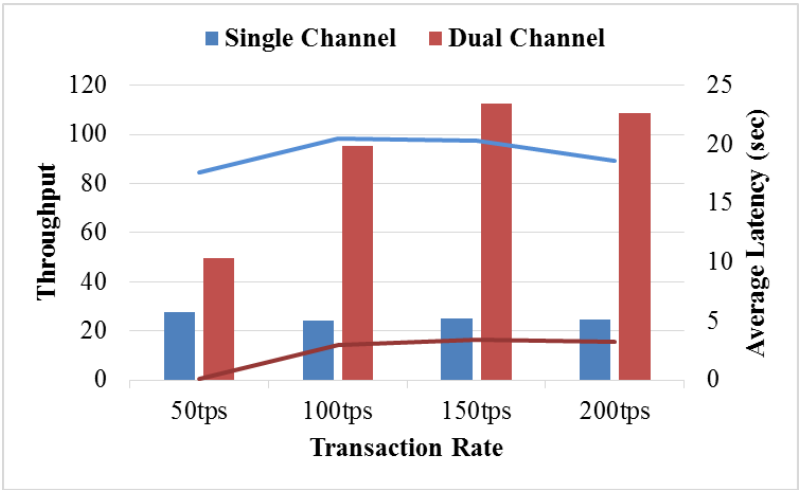


Figure 6.11: Impact of number of channels on Throughput (TPS) and Average Latency (sec)

Scenario 4: Comparative Performance Analysis with other relevant research:

The performance comparison among the existing work [86], [119] and the suggested work is presented in Table 6.2. It presents the performance comparison based on the number of transactions completed in relation to time (sec). Table 6.2 clearly demonstrates that the number of transactions completed, considering both write and all transactions w.r.t time, has increased by three times from the existing work, making the system capable of handling the higher load.

Table 6.2: Comparative Performance Analysis based on no. of transaction completed in relation to time (sec)

Method → Ref. ↓	**No. of write Transactions completed w.r.t. time(sec)**	**No. of all Transactions completed w.r.t. time (sec)**
Tanwar et al. [86]	500, 1000, 1000, 2000, 2000, 3000 (approx.) at time (in secs) 40,60,80,100,120,140 respectively.	Not Performed
Kaur et al. [119]	618, 968, 1354, 1825, 2253, 2810 (approx.) at time (in secs) 40, 60, 80, 100, 120, 140 respectively	669, 979, 1238, 1783, 2049 at the time (in secs) 40, 60, 80, 100, and 120, respectively.
Proposed work	1983,2990, 2078, 2785, 3012, 2860 (approx.) at time (in secs) 40, 60, 80, 100, 120, 140 respectively	2001, 2098, 4000, 5000, 6001, 7001 (approx.) at time (in secs) 40, 60, 80, 100, 120, 140 respectively.

c) Security Analysis of the Designed Framework

The proposed approach implements a dual-channel blockchain-based system integrating two cryptographic algorithms, RSA and AES, for managing healthcare data. In this section, the security analysis of the proposed framework is done to verify its resilience to various types of attacks and provide security features.

- *Sybil Attack:* This is a type of attack in the P2P network where a node creates numerous fraud identities to gain administrative power, control the network and perform malicious activities. The proposed framework is resilient to such attacks as the designed network is authorized. Only authorized and verified individuals can participate in the network.

- *Smart Contract Exploitation Attack*: In this attack, the attacker finds loopholes or vulnerabilities in the designed smart contract that contains the application logic. In our work, the designed SCs are secure and

resistant to exploitation because they are checked for vulnerabilities using the Hyperledger Labs Chaincode Analyzer tool.

- *Data Integrity and Immutability:* Using blockchain technology helps achieve data integrity and immutability. Since each transaction is recorded in the ledger, it cannot be further modified and deleted; it helps in the audit trail and prevents any entity from being denied for their actions performed in the future, achieving a non-repudiation feature.

- *Data confidentiality:* The proposed system provides data confidentiality. As patient data is encrypted using the patient's RSA public key and can only be decrypted by the patient with their corresponding private key, ensuring that only authorized entities can access the confidential information. Similarly, the patient's medical reports are stored and shared using private data collection, further enhancing data confidentiality.

- *Access Control:* In the proposed work, smart contracts and access control lists are utilized to enforce access policies. Such that only authorized entities can perform assigned tasks. For instance, doctors can create and encrypt patients' records, and only lab technicians can add lab reports.

6.4 Chapter Summary

In this chapter, the framework is developed to address the existing security and privacy challenges in managing healthcare data while improving querying capabilities. In the proposed framework, the blockchain network with dual channel and private data collection features is designed and implemented with the consideration of various healthcare entities. Two cryptographic algorithms, RSA and AES, are employed to secure the storage and sharing of EHR data. The access control policies are employed using smart contracts and Access Control Lists (ACLs) to ensure that only permitted entities can access the medical information

and thereby protecting the system from malicious activities. In the end, the experimental evaluation is done to check the system's performance considering transaction sizes, transaction rates, number of channels and rate controllers along with security analysis. The results show throughput increases from 25 to 112.7 TPS at a transaction rate of 150 in dual channel, making the system more efficient as compared to a single channel. Similarly, the latency drops from 20.33 to 3.33 seconds, significantly improving the system performance. Finally, it is concluded that the proposed approach is highly capable of preserving security and privacy standards while also assuring efficient management and accessibility of data in healthcare applications.

Chapter 7

CONCLUSION AND FUTURE DIRECTIONS

This chapter provides the contributions and conclusions regarding the research conducted on the integration of blockchain technology in the healthcare sector for storing, securing, and querying healthcare data. Additionally, it discusses some open issues that can be explored in future research. The research contributions and their outcomes are discussed in Section 7.1, and future directions in this area are presented in Section 7.2

7.1 Conclusion

The main aim of this work is to design and implement a framework for storing, securing and querying healthcare data using blockchain. Blockchain technology is known for its innovative structure, i.e., a cryptographically linked chain of blocks within a cryptographic system. It offers high levels of security, data integrity, and confidentiality while also supporting authentication, authorization, as well as non-repudiation for applications requiring real-time processing. These essential features are often not adequately provided by centralized systems. As a result, blockchain technology emerges as a robust solution for delivering secure and reliable services to decentralized networks, particularly in the healthcare sector. This research thoroughly investigates various blockchain tools and techniques, providing a comprehensive understanding of their applicability in the healthcare sector. It identifies suitable technologies, such as Hyperledger Fabric, that offer the flexibility, scalability, and security needed for managing healthcare data.

The research conducts an extensive review of current blockchain-based EHR systems. This review summarizes research work, and identifies research gaps, challenges faced and possible improvements that can be further considered as future directions.

To address some of the research gaps, a framework is proposed and developed. This framework is developed in three stages. In the first stage, a permissioned blockchain-based architecture involving multiple healthcare entities to store, share, and query EHR data is designed and implemented. Further, the algorithms in terms of smart contracts are defined to describe the functionality of various entities like patients, doctors, TPA, etc. These algorithms utilized the chain code concept to construct an access control mechanism for exchanging records, enabling a complete view of patient medical history. In addition to this, for storing unstructured medical data and for handling rich queries, a NoSQL document-oriented database i.e, CouchDB as a state database, is used. Also, a distributed off-chain storage, IPFS is used to improve the scalability of the proposed framework by storing massive amounts of data off-chain and their corresponding hash values on the Blockchain. Consequently, it eradicates the central authority and mitigates the risk associated with the single point of failure. The integrity of the system is ensured by a tamper-proof, immutable ledger which prevents unauthorized users from altering it. *Hyperledger Caliper* tool is utilized to evaluate the performance of the system for different configurations based on transaction rates and several transactions. The experimental findings show that throughput decreases slightly with the increased peer size during simulation. Thus, the system is scalable, and further performance can be improved by tuning parameters. Further, the system is also evaluated for the handling of complex queries with examples. Also, the comparison between different types of transactions is done by varying batch times. There is a significant increase in throughput and a decrease in latency for both active and query transactions by varying the batch time.

During the second stage, the IB-PRE algorithm is utilized to improve the privacy and security of the healthcare data. The implemented IB-PRE algorithm ensures the secure sharing of records while preserving privacy by avoiding the exposure of private keys. Further, the IB-PRE algorithm is used to share EHRs in a safe manner, where a proxy node retrieves requested data from IPFS, performs re-encryption, and returns the re-encrypted data to the requester, thus ensuring user privacy and data integrity.

In the third stage, the framework implements a dual-channel architecture combined with two cryptographic algorithms, i.e. RSA and AES, to provide the security and rapid retrieval of healthcare information. These encryption techniques deliver safe data transmission via RSA and efficient data storage via AES, offering a secure mechanism to prevent unauthorized access and data breaches. In addition, the concept of private data collection is incorporated to securely store confidential patient information, guaranteeing privacy, security and limited access. Also, an ACL is defined for different users to implement access permissions, i.e., grant and revoke access to viewers while sharing information. This design facilitates efficient data management as well as confidential and secure communication.

Finally, the proposed framework is tested and validated by conducting a qualitative assessment of the designed contracts to ensure proper functionality and interaction among various components. The system's performance is evaluated for different scenarios, including varied transaction loads, transaction types, number of channels, etc. Experimental results indicate that throughput increases from 25 to 112.7 TPS at a transaction rate of 150 in dual channel, making the system more efficient as compared to a single channel. Similarly, the latency drops from 20.33 to 3.33 seconds, significantly improving the system performance. Two vulnerability assessment tools, namely *Chaincode Analyzer* and *Oyente*, are utilized for conducting security analysis of the proposed framework. In addition,

the effectiveness and efficiency of the proposed framework are assessed and compared with existing related works. The evaluation metrics include security attack resistance, features offered, scalability, transaction throughput, and latency. In conclusion, the proposed framework is highly capable of preserving security and privacy standards while also assuring efficient management and accessibility of data in healthcare applications.

7.2 Future Directions

The research work described in this thesis opens up several promising future directions for further research. The proposed framework can be leveraged in various applications beyond the immediate scope of this study. Here are some of the notable applications based on the future directions outlined:

Making the technology accessible to healthcare providers and administrators without any technical expertise can be accomplished through user-friendly interfaces. This application is indispensable in practically employing blockchain in day-to-day healthcare tasks such as appointment scheduling, access to patient records, and real-time health monitoring, among others, for an uninterrupted user experience.

Advanced privacy techniques, including homomorphic encryption and differential privacy, can be used to further enhance the security of applications in research and clinical trials where patient confidentiality and data integrity are paramount.

Integrating Blockchain's security features and AI/ML capabilities can improve healthcare by facilitating early diagnosis of diseases, tailoring treatment strategies, and employing predictive analytics for health outcomes.

Incentive-based approaches can be employed to promote health data sharing and wellness programs by rewarding participants with tokens or other incentives and

supporting public health initiatives.

References

[1] J. Henry, Y. Pylypchuk, T. Searcy, and V. Patel, "Adoption of electronic health record systems among us non-federal acute care hospitals: 2008–2015," *ONC data brief*, vol. 35, pp. 1–9, 2016.

[2] HealthIT.gov, "National trends in hospital and physician adoption of electronic health records," 2021. [Online]. Available: https://www.healthit.gov/data/quickstats/adoption-electronic-health-records-hospital-service-type-2019-2021

[3] A. Saha, R. Amin, S. Kunal, S. Vollala, and S. K. Dwivedi, "Review on blockchain technology based medical healthcare system with privacy issues," *Security and Privacy*, vol. 2, no. 5, pp. 1–14, 2019.

[4] N. P. V. Sravan, P. K. Baruah, S. S. Mudigonda, and P. K. K, "Use of blockchain technology in integrating heath insurance company and hospital," *International Journal of Scientific & Engineering Research*, vol. 9, no. 10, pp. 1664–1669, 2018.

[5] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, "Blockchain-based medical records secure storage and medical service framework," *Journal of medical systems*, vol. 43, no. 1, pp. 1–9, 2019.

[6] "Healthcare data breach report 2019," 2020. [Online]. Available: https://www.hipaajournal.com/2019-healthcare-data-breach-report/s

[7] N. J. Palatty, "80+ healthcare data breach statistics 2024," 2024. [Online]. Available: https://www.getastra.com/blog/security-audit/healthcare-data-breach-statistics

[8] D. Arora, S. Gupta, and A. Anpalagan, "Evolution and adoption of next generation iot-driven health care 4.0 systems," *Wireless Personal Communications*, vol. 127, no. 4, pp. 3533–3613, 2022.

[9] F. Xhafa, J. Feng, Y. Zhang, X. Chen, and J. Li, "Privacy-aware attribute-based phr sharing with user accountability in cloud computing," *The Journal of Supercomputing*, vol. 71, no. 5, pp. 1607–1619, 2015.

[10] S. A. Soleymani, S. Goudarzi, M. H. Anisi, A. Jindal, N. Kama, and S. A. Ismail, "A privacy-preserving authentication scheme for real-time medical monitoring systems," *IEEE journal of biomedical and health informatics*, vol. 27, no. 5, pp. 2314–2322, 2022.

[11] J. J. Rodrigues, I. de la Torre, G. Fernández, and M. López-Coronado, "Analysis of the security and privacy requirements of cloud-based electronic health records systems," *Journal of medical Internet research*, vol. 15, no. 8, p. e186, 2013.

[12] A. Sinha, S. Singh, and H. K. Verma, "Ai-driven task scheduling strategy with blockchain integration for edge computing," *Journal of Grid Computing*, vol. 22, no. 13, pp. 1–16, 2024.

[13] P. Kamboj, S. Kumar, and V. Goyal, "Measuring and mitigating gender bias in contextualized word embeddings," in *Proceedings of IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS-23)*, New Raipur, India, pp. 1–5,October 2023.

[14] Robmenzies, "Healthcare data management meet blockchain." [Online]. Available: https://steemit.com/healthcare/@robmenzies/healthcare-data-management-meet-blockchain

[15] M. K. Pratama FA, "Query support for data processing and analysis on ethereum blockchain," in *International Symposium on Electronics and Smart Devices (ISESD-18)*, Bandung, Indonesia, pp. 1–5, October 2018.

[16] L. Chen, W.-K. Lee, C.-C. Chang, K.-K. R. Choo, and N. Zhang, "Blockchain based searchable encryption for electronic health record sharing," *Future Generation Computer Systems*, vol. 95, pp. 420–429, 2019.

[17] T. Tegegne and T. P. T. van der Weide, "Enriching queries with user preferences in healthcare," *Information processing & management*, vol. 50, no. 4, pp. 599–620, 2014.

[18] S. Makani, R. Pittala, E. Alsayed, M. Aloqaily, and Y. Jararweh, "A survey of blockchain applications in sustainable and smart cities," *Cluster Computing*, vol. 25, pp. 3915–3936, 2022.

[19] V. Buterin, "A next-generation smart contract and decentralized application platform," 2014. [Online]. Available: https://ethereum.org/en/whitepaper/

[20] W. She, J.-s. Chen, Q. Liu, Y. Hu, Z. Gu, Z. Tian, and W. Liu, "New Blockchain Technology for Medical Big Data Security Sharing," *Journal of Chinese Computer System*, vol. 40, no. 7, pp. 1449–1454, 2019.

[21] G. Iredale, 2020. [Online]. Available: https://101blockchains.com/history-of-blockchain-timeline/

[22] S. Nakamoto and A. Bitcoin, "A peer-to-peer electronic cash system," *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf*, vol. 4, no. 2, pp. 1–5, 2008.

[23] NPTEL, 2018. [Online]. Available: http://textofvideo.nptel.ac.in/106105184/lec1.pdf

[24] M. Iansiti and K. R. Lakhani, "The truth about blockchain," *Harvard Business Review*, vol. 95, no. 1, pp. 118–127, 2017.

[25] S. Haber and W. S. Stornetta, "How to time-stamp a digital document," *Journal of Cryptology*, vol. 3, no. 2, pp. 99–111, 1991.

[26] DataFlair, "Understanding the basics of blockchain–nourish the roots of technology," 2019. [Online]. Available: https://data-flair.training/blogs/basics-of-blockchain-technology/

[27] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[28] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and Open Research Challenges," *IEEE Access*, vol. 7, pp. 10 127–10 149, 2019.

[29] "Tendermint,2019," 2020. [Online]. Available: https://docs.tendermint.com/master/introduction/what-is-tendermint.html

[30] L. Baird, B. Gross, and D. Thibeau, "Hedera Consensus Service," WhitePaper, 2019. [Online]. Available: https://files.hedera.com/hh-consensus-service-whitepaper.pdf?

[31] V. Buterin, "What is ethereum? ethereum official webpage," 2020. [Online]. Available: http://www.ethdocs.org/en/latest/introduction/what-is-ethereum.html

[32] M. Xu, X. Chen, and G. Kou, "A systematic review of blockchain," *Financial innovation*, vol. 5, no. 1, pp. 1–14, 2019.

[33] S. K. Singh, V. Tiwari, and V. R. Vadi, "Smart Contract Using Solidity (Remix–Ethereum IDE)," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 12, no. 2, pp. 243–249, 2023.

[34] S. C. Team, "Truffle in Blockchain: Revolutionizing DApp DevelopmentShardeum — shardeum.org," 2024. [Online]. Available: https://shardeum.org/blog/truffle/

[35] J. Chittoda, *Mastering Blockchain Programming with Solidity: Write production-ready smart contracts for Ethereum blockchain with Solidity*. Packt Publishing Ltd, 2019.

[36] L. V.-C. Thuy, K. Cao-Minh, C. Dang-Le-Bao, and T. A. Nguyen, "Votereum: An Ethereum-Based E-Voting System," in *Proceedings of IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF-19)*, Danang, Vietnam, pp. 1–6, March 2019.

[37] L. Zhou, X. Xiong, J. Ernstberger, S. Chaliasos, Z. Wang, Y. Wang, K. Qin, R. Wattenhofer, D. Song, and A. Gervais, "SoK: Decentralized Finance (DeFi) Attacks," in *Proceedings of IEEE Symposium on Security and Privacy (SP)*, Cambridge MA USA, pp. 2444–2461, September 2023.

[38] C. Catalini and J. S. Gans, "Some simple economics of the blockchain," *Communications of the ACM*, vol. 63, no. 7, pp. 80–90, 2020.

[39] K. R. Ludvigsen, S. Nagaraja, and A. Daly, "Preventing or mitigating adversarial supply chain attacks: A legal analysis," in *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*, Los Angeles CA USA, pp. 25–34, November 2022.

[40] H. Treiblmaier, "The impact of the blockchain on the supply chain: a theory-based research framework and a call for action," *Supply Chain Management: An International Journal*, vol. 23, no. 6, pp. 545–559, 2018.

[41] N. Kshetri, "1 Blockchain's roles in meeting key supply chain management objectives," *International Journal of Information Management*, vol. 39, pp. 80–89, 2018.

[42] M. M. Schöner, D. Kourouklis, P. Sandner, E. Gonzalez, and J. Förster, "Blockchain technology in the pharmaceutical industry," *Frankfurt, Germany: Frankfurt School Blockchain Center*, pp. 1–9, 2017.

[43] Feng Tian, "An agri-food supply chain traceability system for China based on RFID & blockchain technology," in *Proceedings of IEEE 13th International Conference on Service Systems and Service Management (ICSSSM-16)*, Kunming, China, pp. 1–6, June 2016.

[44] Mohit, S. Kaur, and M. Singh, “Design and implementation of blockchain-based supply chain framework with improved traceability, privacy, and ownership,” *Cluster Computing*, vol. 27, no. 3, pp. 2345–2363, 2024.

[45] H. Min, “Blockchain technology for enhancing supply chain resilience,” *Business Horizons*, vol. 62, no. 1, pp. 35–45, 2019.

[46] V. Charles, A. Emrouznejad, and T. Gherman, “A critical analysis of the integration of blockchain and artificial intelligence for supply chain,” *Annals of Operations Research*, vol. 327, no. 1, pp. 7–47, 2023.

[47] A. Sharma, S. Bahl, A. K. Bagha, M. Javaid, D. K. Shukla, and A. Haleem, “Blockchain technology and its applications to combat COVID-19 pandemic,” *Research on Biomedical Engineering*, vol. 38, no. 1, pp. 173–180, 2022.

[48] T.-T. Kuo, H.-E. Kim, and L. Ohno-Machado, “Blockchain distributed ledger technologies for biomedical and health care applications,” *Journal of the American Medical Informatics Association*, vol. 24, no. 6, pp. 1211–1220, 2017.

[49] R. W. Ahmad, K. Salah, R. Jayaraman, I. Yaqoob, S. Ellahham, and M. Omar, “The role of blockchain technology in telehealth and telemedicine,” *International Journal of Medical Informatics*, vol. 148, p. 104399, 2021.

[50] M. Biletic, F. H. Juwono, and L. Gopal, “Nanonetworks and molecular communications for biomedical applications,” *IEEE Potentials*, vol. 39, no. 3, pp. 25–30, 2020.

[51] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, “MedRec: Using Blockchain for Medical Data Access and Permission Management,” in *Proceedings 2nd International Conference on Open and Big Data (OBD)*, Vienna, Austria, pp. 25–30, August 2016.

[52] W. J. Gordon and C. Catalini, "Blockchain Technology for Healthcare: Facilitating the Transition to Patient-Driven Interoperability," *Computational and Structural Biotechnology Journal*, vol. 16, pp. 224–230, 2018.

[53] M. Mettler, "Blockchain technology in healthcare: The revolution starts here," in *Proceedings of IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Munich, Germany, pp. 1–3, September 2016.

[54] A. Al Omar, M. S. Rahman, A. Basu, and S. Kiyomoto, "MediBchain: A Blockchain Based Privacy Preserving Platform for Healthcare Data," in *Proceedings of International conference on security, privacy and anonymity in computation, communication and storage.*, Guangzhou, China, pp. 534–543, December 2017.

[55] J. Biggs, "Sierra Leone just ran the first blockchain-based election." 2018. [Online]. Available: https://techcrunch.com/2018/03/14/sierra-leone-just-ran-the-first-blockchain-based-election/

[56] A. Shah, N. Sodhia, S. Saha, S. Banerjee, and M. Chavan, "Blockchain Enabled Online-Voting System," *ITM Web of Conferences*, vol. 32, pp. 1–6, 2020.

[57] P. Schmidt, "Blockcerts—an open infrastructure for academic credentials on the blockchain," *MIT Media Lab*, 2016. [Online]. Available: https://medium.com/mit-media-lab/blockcerts-an-open-infrastructure-for\-academic-credentials-on-the-blockchain-899a6b880b2f

[58] S. Rasool, A. Saleem, M. Iqbal, T. Dagiuklas, S. Mumtaz, and Z. U. Qayyum, "Docschain: Blockchain-Based IoT Solution for Verification of Degree Documents," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 3, pp. 827–837, Jun. 2020.

[59] K. Madhura and R. Mahalakshmi, “Usage of block chain in real estate business for transparency and improved security,” in *Proceedings of International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI-22)*, Chennai, India, pp. 1–10, January 2022.

[60] S. Aggarwal, R. Chaudhary, G. S. Aujla, N. Kumar, K.-K. R. Choo, and A. Y. Zomaya, “Blockchain for smart communities: Applications, challenges and opportunities,” *Journal of Network and Computer Applications*, vol. 144, pp. 13–48, 2019.

[61] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach, D. Jenkins, P. McCallum, and A. Peacock, “Blockchain technology in the energy sector: A systematic review of challenges and opportunities,” *Renewable and Sustainable Energy Reviews*, vol. 100, pp. 143–174, 2019.

[62] E. Mengelkamp, B. Notheisen, C. Beer, D. Dauer, and C. Weinhardt, “A blockchain-based smart grid: towards sustainable local energy markets,” *Computer Science-Research and Development*, vol. 33, no. 1-2, pp. 207–214, 2018.

[63] E. Mengelkamp, J. Gärttner, K. Rock, S. Kessler, L. Orsini, and C. Weinhardt, “Designing microgrid energy markets,” *Applied Energy*, vol. 210, pp. 870–880, 2018.

[64] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, and D. I. Kim, “A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks,” *IEEE Access*, vol. 7, pp. 22 328–22 370, 2019.

[65] A. Lei, H. Cruickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, “Blockchain-based dynamic key management for heterogeneous intelligent transportation systems,” *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1832–1843, 2017.

[66] Q. Zhang, Y. He, R. Lai, Z. Hou, and G. Zhao, "A survey on the efficiency, reliability, and security of data query in blockchain systems," *Future Generation Computer Systems*, vol. 145, pp. 303–320, 2023.

[67] J. Gao, J. Zhang, Y. Li, J. Hao, K. Wang, Z. Guan, and Z. Chen, "Chaindb: Ensuring integrity of querying off-chain data on blockchain," in *Proceedings of 5th International Conference on Blockchain Technology and Applications*, pp. 175–181, December 2022.

[68] M. S. Rahman, I. Khalil, N. Moustafa, A. P. Kalapaaking, and A. Bouras, "A blockchain-enabled privacy-preserving verifiable query framework for securing cloud-assisted industrial internet of things systems," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 5007–5017, 2021.

[69] M. Kaandorp, "Easy and efficient querying of smart contract data while maintaining data integrity," *M.Sc thesis, Universiteit van Amsterdam*, pp. 1–44, July 2019. [Online]. Available: https://staff.fnwi.uva.nl/aszbelloum/MSctheses/MScthesis_Matthijs_Kaandorp.pdf

[70] N. Iqbal, F. Jamil, S. Ahmad, and D. Kim, "A novel blockchain-based integrity and reliable veterinary clinic information management system using predictive analytics for provisioning of quality health services," *IEEE Access*, vol. 9, pp. 8069–8098, 2021.

[71] S. Ma and X. Zhang, "Integrating blockchain and zk-rollup for efficient healthcare data privacy protection system via ipfs," *Scientific Reports*, vol. 14, no. 1, p. 11746, 2024.

[72] J. Jayabalan and N. Jeyanthi, "Scalable blockchain model using off-chain ipfs storage for healthcare data security and privacy," *Journal of Parallel and distributed computing*, vol. 164, pp. 152–167, 2022.

[73] J. Partala, T. H. Nguyen, and S. Pirttikangas, "Non-interactive

zero-knowledge for blockchain: A survey," *IEEE Access*, vol. 8, pp. 227 945–227 961, 2020.

[74] S. Capraz and A. Ozsoy, "Personal data protection in blockchain with zero-knowledge proof," *Blockchain Technology and Innovations in Business Processes*, vol. 219, pp. 109–124, 2021.

[75] N. Sangeeta and S. Y. Nam, "Blockchain and interplanetary file system (ipfs)-based data storage system for vehicular networks with keyword search capability," *Electronics*, vol. 12, no. 7, p. 1545, 2023.

[76] A. Singh, K. Sharma, and P. K. Sarangi, "Vulnerabilities in smart contracts of decentralized blockchain," in *Proceedings of International Conference on Recent Developments in Cyber Security*, vol. 896, pp. 551–566,2023.

[77] V. K. Jain and M. Tripathi, "An integrated deep learning model for ethereum smart contract vulnerability detection," *International Journal of Information Security*, vol. 23, no. 1, pp. 557–575, 2024.

[78] K. Kaur, S. Tomar, and M. Tripathi, "Gas fee reduction by detecting loop fusible patterns in ethereum smart contract," in *Proceedings of IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS-22)*, Gandhinagar, Gujarat, India, pp. 458–463, December 2022.

[79] B. Arunkumar and G. Kousalya, "Blockchain-based decentralized and secure lightweight e-health system for electronic health records," in *Proceedings of Fifth Intelligent Systems, Technologies and Applications (ISTA-2019), India*, vol. 1148, pp. 273–289, May 2020.

[80] K. Fan, S. Wang, Y. Ren, H. Li, and Y. Yang, "Medblock: Efficient and secure medical data sharing via blockchain," *Journal of medical systems*, vol. 42, no. 8, p. 136, 2018.

[81] L. Ismail, H. Materwala, and S. Zeadally, "Lightweight blockchain for healthcare," *IEEE Access*, vol. 7, pp. 149 935–149 951, 2019.

[82] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani, "Medshare: Trust-less medical data sharing among cloud service providers via blockchain," *IEEE Access*, vol. 5, pp. 14 757–14 767, 2017.

[83] J. Vora, A. Nayyar, S. Tanwar, S. Tyagi, N. Kumar, M. S. Obaidat, and J. J. P. C. Rodrigues, "Bheem: A blockchain-based framework for securing electronic health records," in *2018 IEEE Globecom Workshops (GC Wkshps)*, Abu Dhabi, United Arab Emirates, pp. 1–6, December 2018.

[84] G. Yang and C. Li, "A Design of Blockchain-Based Architecture for the Security of Electronic Health Record (EHR) Systems," in *Proceedings of International Conference on Cloud Computing Technology and Science (CloudCom)* , Nicosia, Cyprus, pp. 261–265, December 2018.

[85] H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, and S. Liu, "Blockchain-based data preservation system for medical data," *Journal of medical systems*, vol. 42, no. 8, pp. 1–13, 2018.

[86] S. Tanwar, K. Parekh, and R. Evans, "Blockchain-based electronic healthcare record system for healthcare 4.0 applications," *Journal of Information Security and Applications*, vol. 50, pp. 1–13, 2020.

[87] P. Bhattacharya, S. Tanwar, U. Bodke, S. Tyagi, and N. Kumar, "Bindaas: Blockchain-based deep-learning as-a-service in healthcare 4.0 applications," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1242 – 1255, 2019.

[88] A. S. Koushik, B. Jain, N. Menon, D. Lohia, S. Chaudhari, and V. K. B.P, "Performance analysis of blockchain-based medical records management system," in *Proceedings of IEEE 4th International Conference on Recent Trends*

on Electronics, Information, Communication Technology (RTEICT-19), pp. 985–989, May 2019.

[89] A. Roehrs, C. A. da Costa, R. da Rosa Righi, V. F. da Silva, J. R. Goldim, and D. C. Schmidt, "Analyzing the performance of a blockchain-based personal health record implementation," *Journal of biomedical informatics*, vol. 92, pp. 1–9, 2019.

[90] G. Yang, C. Li, and K. E. Marstein, "A blockchain-based architecture for securing electronic health record systems," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 14, pp. 1–10, 2019.

[91] L. Hang, E. Choi, and D.-H. Kim, "A novel EMR integrity management based on a medical blockchain platform in hospital," *Electronics*, vol. 8, no. 4, pp. 1–27, 2019.

[92] C. X. T. K. R. Sukhwani H, Martinez JM, "A. performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric)," in *Proceedings of IEEE 36th Symposium on Reliable Distributed Systems (SRDS-17)*, Hong Kong, China, pp. 253–255, September 2017.

[93] A. K. Walzade and P. A. Vikhar, "Secured Electronic Clinical Unified Record Exchange using Blockchain Ledger and Optimized Cryptography," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 12s, pp. 46–59, 2024.

[94] J. Kaur, R. Rani, and N. Kalra, "Attribute-based access control scheme for secure storage and sharing of EHRs using blockchain and IPFS," *Cluster Computing*, vol. 27, pp. 1047–1061, 2023.

[95] V. Koufi, F. Malamateniou, and G. Vassilacopoulos, "Ubiquitous access to cloud emergency medical services," in *Proceedings of the 10th IEEE International Conference on Information Technology and Applications in Biomedicine*, Corfu, Greece, pp. 1–4, November 2010.

[96] J. Cai, H. Huang, C. Ma, and J. Liu, "A blockchain-based privacy protecting framework with multi-channel access control model for asset trading," *Peer-to-Peer Networking and Applications*, pp. 1–20, 2024.

[97] Díaz and H. Kaschel, "Scalable Electronic Health Record Management System Using a Dual-Channel Blockchain Hyperledger Fabric," *Systems*, vol. 11, no. 7, pp. 1–23, Jul. 2023.

[98] S. Vidhya and V. Kalaivani, "A blockchain based secure and privacy aware medical data sharing using smart contract and encryption scheme," *Peer-to-Peer Networking and Applications*, vol. 16, no. 2, pp. 900–913, Mar. 2023.

[99] A. I. Sanka and R. C. Cheung, "A systematic review of blockchain scalability: Issues, solutions, analysis and future research," *Journal of Network and Computer Applications*, vol. 195, no. 1, pp. 1–25, 2021.

[100] M. Ndzimakhwe, A. Telukdarie, I. Munien, A. Vermeulen, U. K. Chude-Okonkwo, and S. P. Philbin, "A framework for user-focused electronic health record system leveraging hyperledger fabric," *Information*, vol. 14, no. 1, pp. 1–25, 2023.

[101] P. Sharma, R. Jindal, and M. D. Borah, "Blockchain-based cloud storage system with cp-abe-based access control and revocation process," *The Journal of Supercomputing*, vol. 78, pp. 7700—7728, 2022.

[102] A. Ali, M. A. Almaiah, F. Hajjej, M. F. Pasha, O. H. Fang, R. Khan, J. Teo, and M. Zakarya, "An industrial iot-based blockchain-enabled secure searchable encryption approach for healthcare systems using neural network," *Sensors*, vol. 22, no. 2, pp. 1–20, 2022.

[103] A. Ali, H. A. Rahim, J. Ali, M. F. Pasha, M. Masud, A. U. Rehman, C. Chen, and M. Baz, "A novel secure blockchain framework for accessing electronic

health records using multiple certificate authority," *Applied Sciences*, vol. 11, no. 21, pp. 1–26, 2021.

[104] F. Li, K. Liu, L. Zhang, S. Huang, and Q. Wu, "EHRChain: A Blockchain-Based EHR System Using Attribute-Based and Homomorphic Cryptosystem," *IEEE Transactions on Services Computing*, vol. 15, no. 5, pp. 2755–2765, 2022.

[105] A. Ali, H. A. Rahim, M. F. Pasha, R. Dowsley, M. Masud, J. Ali, and M. Baz, "Security, privacy, and reliability in digital healthcare systems using blockchain," *Electronics*, vol. 10, no. 16, pp. 20–34, 2021.

[106] A. Mubarakali, "Healthcare services monitoring in cloud using secure and robust healthcare-based blockchain (srhb) approach," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1330–1337, 2020.

[107] N. Saravanan and A. Umamakeswari, "HAP-CP-ABE based encryption technique with hashed access policy based authentication scheme for privacy preserving of PHR," *Microprocessors and Microsystems*, vol. 80, pp. 1–10, 2021.

[108] A. Shahnaz, U. Qamar, and A. Khalid, "Using blockchain for electronic health records," *IEEE Access*, vol. 7, pp. 147 782–147 795, 2019.

[109] T. T. Thwin and S. Vasupongayya, "Blockchain-based access control model to preserve privacy for personal health record systems," *Security and Communication Networks*, vol. 2019, no. 1, pp. 1–15, 2019.

[110] J. Andrew, D. P. Isravel, K. M. Sagayam, B. Bhushan, Y. Sei, and J. Eunice, "Blockchain for healthcare systems: Architecture, security challenges, trends and future directions," *Journal of Network and Computer Applications*, vol. 215, pp. 1–36, 2023.

[111] "Transaction flow,2020," 2020. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.0/txflow.html

[112] Q. Zheng, Y. Li, P. Chen, and X. Dong, "An Innovative IPFS-Based Storage Model for Blockchain," in *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, Santiago, Chile, pp. 704–708, December 2018.

[113] Go, "Documentation - The Go Programming Language." [Online]. Available: https://golang.org/doc/.

[114] R. Zhang, R. Xue, and L. Liu, "Security and privacy for healthcare blockchains," *IEEE Transactions on Services Computing.*, vol. 15, no. 6, pp. 3668–3686, 2021.

[115] S. A. Soleymani, S. Goudarzi, M. H. Anisi, A. Jindal, N. Kama, and S. A. Ismail, "A privacy-preserving authentication scheme for real-time medical monitoring systems," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 5, pp. 2314–2322, 2023.

[116] D. Nuñez, I. Agudo, and J. Lopez, "Proxy re-encryption: Analysis of constructions and its application to secure access delegation," *Journal of Network and Computer Applications.*, vol. 87, pp. 193–209, 2017.

[117] T.-F. Xue, Q.-C. Fu, C. Wang, and X. Wang, "A medical data sharing model via blockchain," *Acta Automatica Sinica.*, vol. 43, no. 9, pp. 1555–1562, 2017.

[118] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Proceedings of International Conference on Applied Cryptography and Network Security.* , vol. 4521,Berlin, Heidelberg, pp. 288–306,2007.

[119] J. Kaur, R. Rani, and N. Kalra, "Blockchain-based framework for secured storage, sharing, and querying of electronic healthcare records," *Concurrency and Computation: Practice and Experience.*, vol. 33, no. 20, pp. 1–24, 2021.

[120] U. Chelladurai and S. Pandian, "A novel blockchain based electronic health record automation system for healthcare," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, pp. 693–703, 2022.

[121] M. U. Chelladurai, D. S. Pandian, and D. K. Ramasamy, "A blockchain based patient centric electronic health record storage and integrity management for e-Health systems," *Health Policy and Technology.*, vol. 10, no. 4, pp. 1–32, 2021.

[122] IBM, "IBMDeveloper-Recipes," 2019. [Online]. Available: https://developer.ibm.com/recipes/tutorials/writing-hyperledger-fabric-chaincodeusing-go-programming-language/

[123] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering.*, vol. 3, no. 2, pp. 111–128, 2013.

[124] K. Yamashita, Y. Nomura, E. Zhou, B. Pi, and S. Jun, "Potential risks of hyperledger fabric smart contracts," in *Proceedings of IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE).*, Hangzhou, China, pp. 1–10, February 2019.

[125] FujitsuLaboratories, "Chaincode Analyzer," 2020. [Online]. Available: https://github.com/FujitsuLaboratories/ChaincodeAnalyzer.

[126] E. Milanov, "The rsa algorithm," *RSA laboratories*, pp. 1–11, 2009.

[127] M. Dworkin, E. Barker, J. Nechvatal, J. Foti, L. Bassham, E. Roback, and J. Dray, "Advanced encryption standard (aes), Federal Inf," *Process. Stds.(NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD*, vol. 11, 2001.

[128] N. A. Ugochukwu, S. Goyal, A. S. Rajawat, S. M. Islam, J. He, and M. Aslam, "An innovative blockchain-based secured logistics

management architecture: utilizing an RSA asymmetric encryption method," *Mathematics*, vol. 10, no. 24, pp. 1–18, 2022.

[129] N. Sammeta and L. Parthiban, "Hyperledger blockchain enabled secure medical record management with deep learning-based diagnosis model," *Complex & Intelligent Systems*, vol. 8, no. 1, pp. 625–640, 2022.

[130] HyperLedger, "HyperLedger Fabric," 2020. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/latest/msp.html

List of Research Publications

Journal (SCI/SCIE)

1. J. Kaur, R. Rani, and N. Kalra, "**Blockchain-based framework for secured storage, sharing, and querying of Electronic Healthcare Records**," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 20, pp. 1–24, 2021. doi:10.1002/cpe.6369 [Impact Factor: 1.5]

2. J. Kaur, R. Rani, and N. Kalra, "**A Blockchain-based framework for Privacy Preservation of Electronic Health Records (EHRs)**," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 9, pp. 1–18, 2022. doi:10.1002/ett.4507 [Impact Factor: 2.5]

3. J. Kaur, R. Rani, and N. Kalra, "**Attribute-based access control scheme for secure storage and sharing of EHRs using blockchain and IPFS**," *Cluster Computing*, vol. 27, pp. 1047-1061, 2023. doi:10.1007/s10586-023-04038-2 [Impact Factor: 3.6]

4. J. Kaur, R. Rani, and N. Kalra, "**Dual Channel Blockchain Based Framework for Healthcare Data**," *Transactions on Emerging Telecommunications Technologies*, 2024. [Impact Factor: 2.5](Major Revision Received)

Conferences

1. J. Kaur, R. Rani and N. Kalra, "**A Blockchain Enabled Predictive, Analytical Model for Fraud Detection in Healthcare Data**," *in Proceedings of IEEE Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC-22)*, Solan, Himachal Pradesh, India, pp. 319-324, November 2022, doi: 10.1109/PDGC56933.2022. 10053120.

2. J. Kaur, R. Rani and N. Kalra, "**An Automated Liver Disease Detection**

System using Machine Learning and Smart Contract," *in Proceedings of IEEE International Conference on Current Development in Engineering and Technology (CCET-22)*, Bhopal, India, pp. 1-5, December 2022, doi: 10.1109/CCET56606.2022.10080177.